

BAB 3

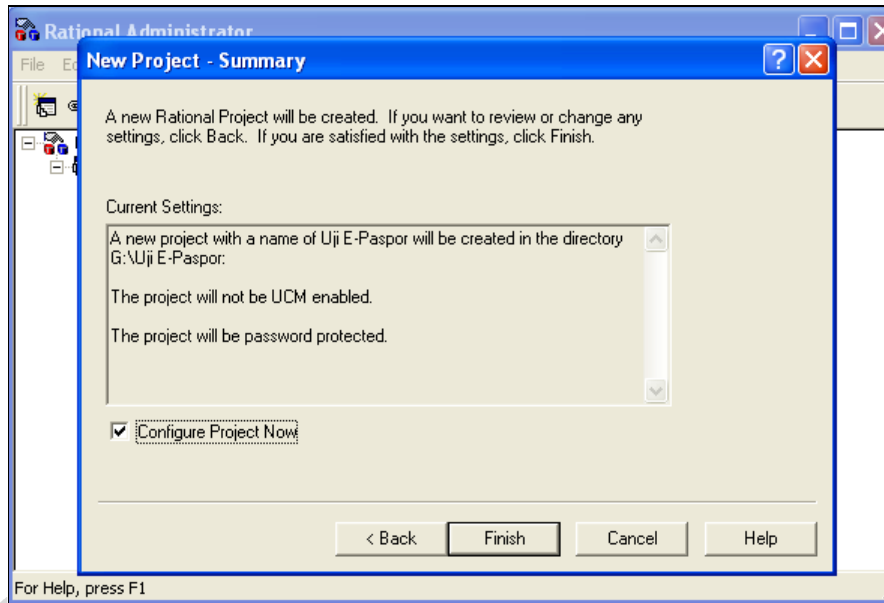
PERENCANAAN

Pada bab ini akan dijelaskan mengenai aktivitas apa saja yang dilakukan pada awal eksperimen, yaitu perencanaan pengujian. Eksperimen ini menggunakan studi kasus aplikasi E-Paspor.

3.1 Persiapan Pengujian

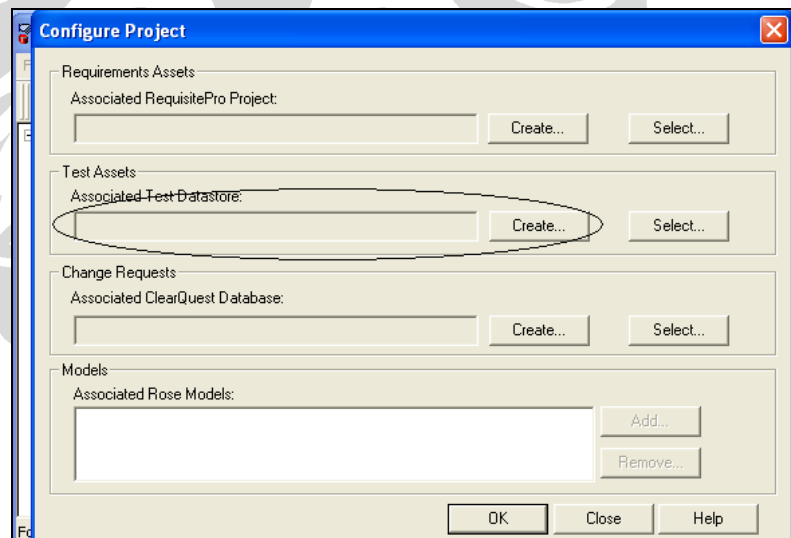
Karena semua alur kerja di atas menggunakan Rational TestManager dan Rational Robot, maka ada baiknya kita mengetahui bagaimana mempersiapkan keduanya agar dapat digunakan dalam pengujian. Sebelum menggunakan komponen-komponen pengujian dengan Rational, maka terlebih dahulu harus dibuat Rational Suite Project (.rsp) dengan menggunakan Rational Administrator. Caranya adalah sebagai berikut:

1. Buka Rational Administrator
2. Pilih *File* → *New Project*
3. Masukkan nama proyek untuk pengujian, dalam hal ini untuk proyek pengujian aplikasi E-Paspor, maka nama proyeknya adalah “Uji E-Paspor” dan lokasi penyimpanannya proyek (bisa langsung membuat *folder* baru)
4. Masukkan *username* dan *password* mengakses, mengatur konfigurasi, atau menghapus proyek ini
5. Pilih *Next*
6. Muncul *New Project – Summary* → pilih *Finish* (pastikan *Configure Project Now* terpilih)



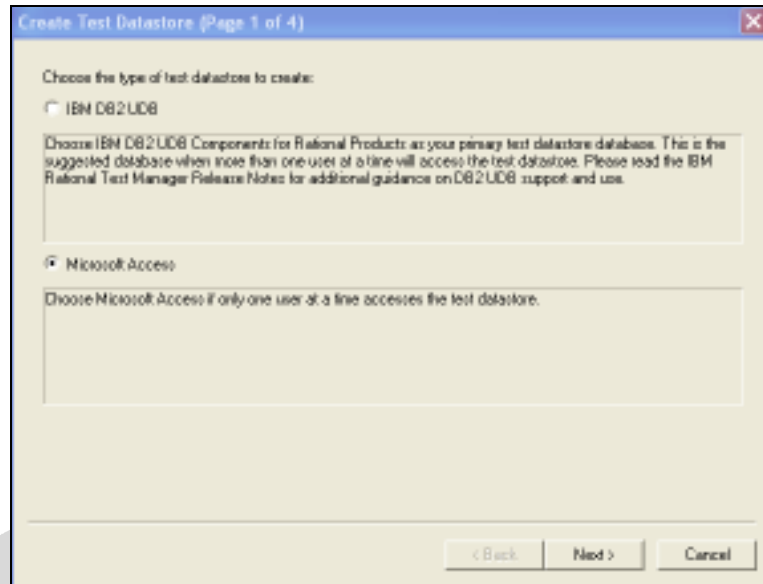
Gambar 3.1 *Window New Project Summary*

7. Pada grup *Test Assets – Associated Test Datastore* → pilih *Create*



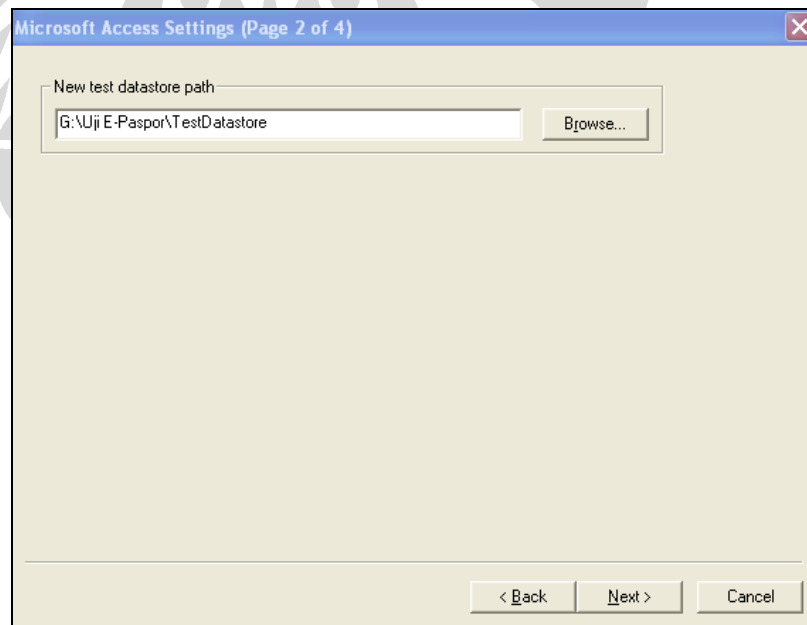
Gambar 3.2 *Create Test Datastore*

8. pilih *Microsoft Access* → *Next*



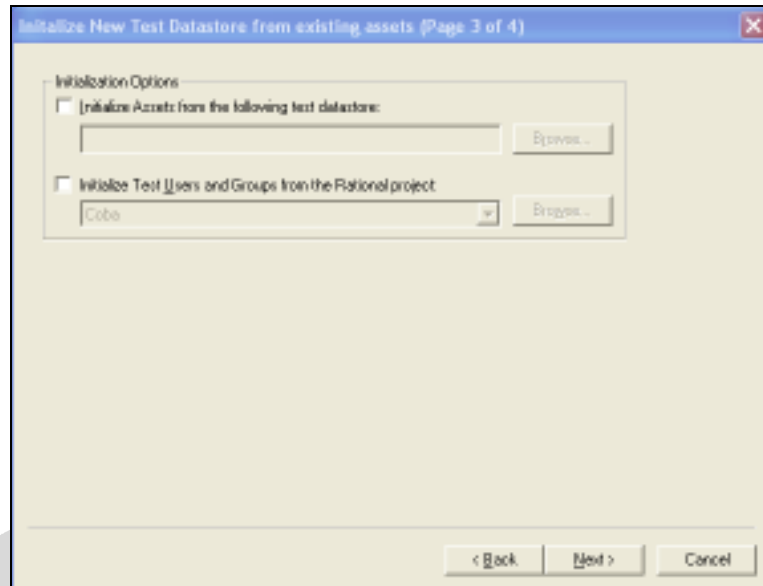
Gambar 3.3 Create Test Datastore dengan Microsoft Access

9. Muncul *Microsoft Access Setting* dan *New test datastore path* → isikan lokasi dimana *datastore* akan disimpan (disarankan untuk mengikuti *default* yang sudah dituliskan) → Next



Gambar 3.4 Microsoft Access Setting

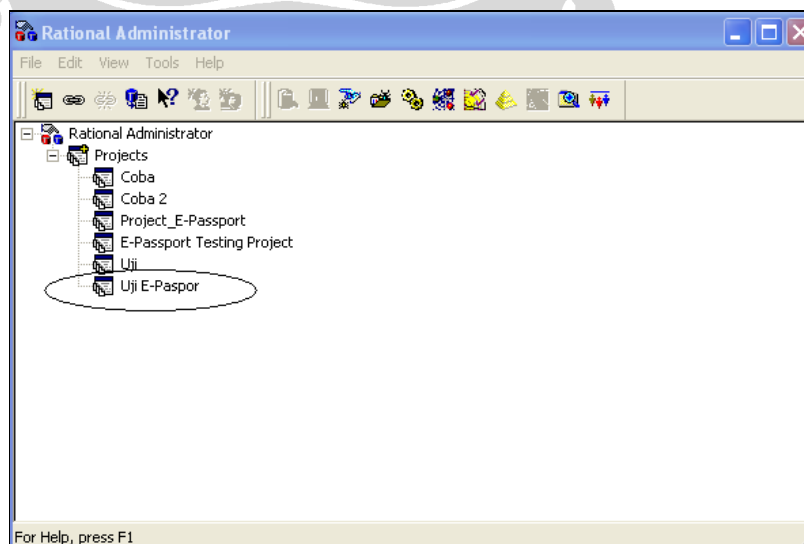
10. Muncul *Initialize New Test Datastore from existing assets* → Next



Gambar 3.5 Initialize New Test Datastore from existing assets

11. Muncul *Create Test Datastore Summary* → Finish

Setelah berhasil membuat *datastore* yang akan menyimpan semua aset pengujian, tutup semua *window* Administrator. Sebelum menutup window utama Rational Administrator, nama Uji E-Paspor telah muncul di daftar Projects seperti yang diperlihatkan pada gambar berikut:



Gambar 3.6 Pembuatan Proyek Rational yang Berhasil

Universitas Indonesia

Gambar 3.6 menunjukkan bahwa pembuatan proyek Rational untuk pengujian berhasil.

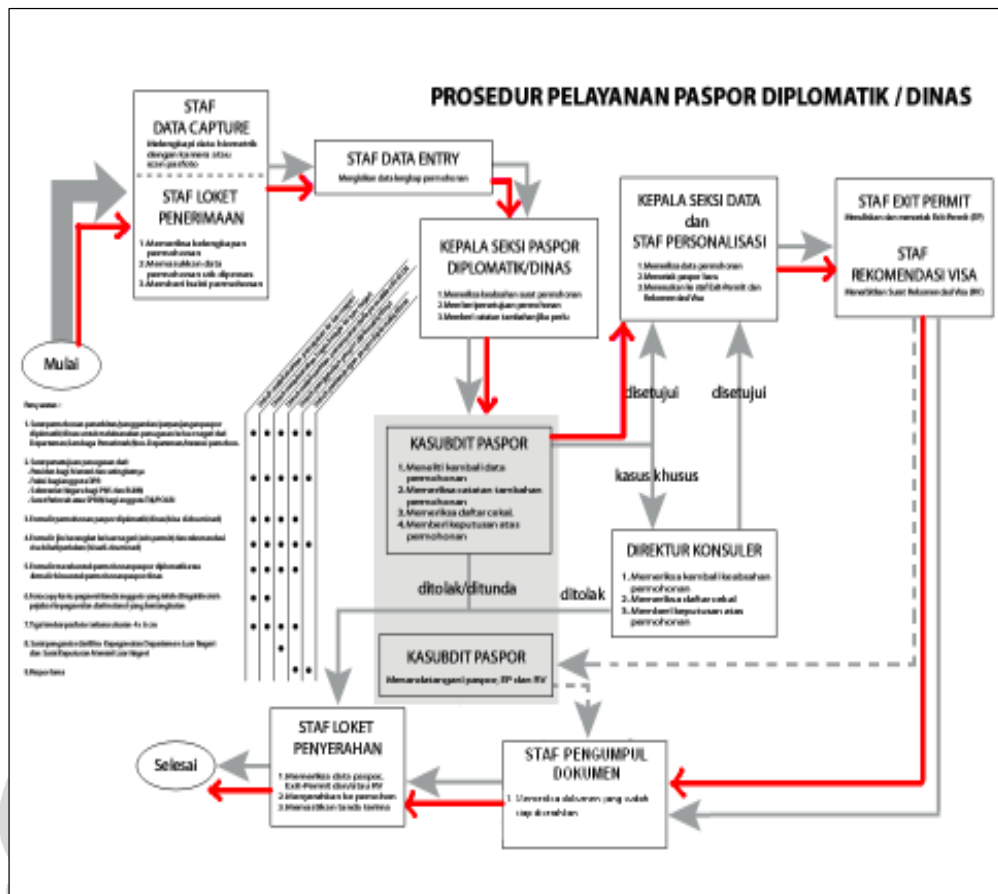
3.2 Perencanaan

Dalam tahap ini, eksperimen dilakukan dengan membentuk *test plan* di Rational TestManager. Pembuatan *test plan* ini disertai dengan pembentukan *test case*. Berdasarkan tutorial *Generating Test Cases From Use Cases* oleh Jim Heumann, *test case* dapat dibentuk dari *use case*. Oleh karena itu, subbab ini akan berisi penjelasan mengenai:

1. Mendaftar semua *use case* dari aplikasi E-Paspor dan menentukan *use case* apa saja yang akan diuji
2. Membentuk *test plan* dari *use case* yang telah ditentukan
3. Membentuk *test case* dari *use case* yang telah ditentukan

3.2.1 Penentuan *use case* pada aplikasi E-Paspor yang akan diuji

Sebelum menentukan *use case* apa saja yang terdapat dalam aplikasi E-Paspor, perlu diketahui prosedur pelayanan paspor di Departemen Luar Negeri, baik secara manual maupun melalui sistem. Prosedur pelayanan paspor secara garis besar dapat dilihat pada gambar berikut:



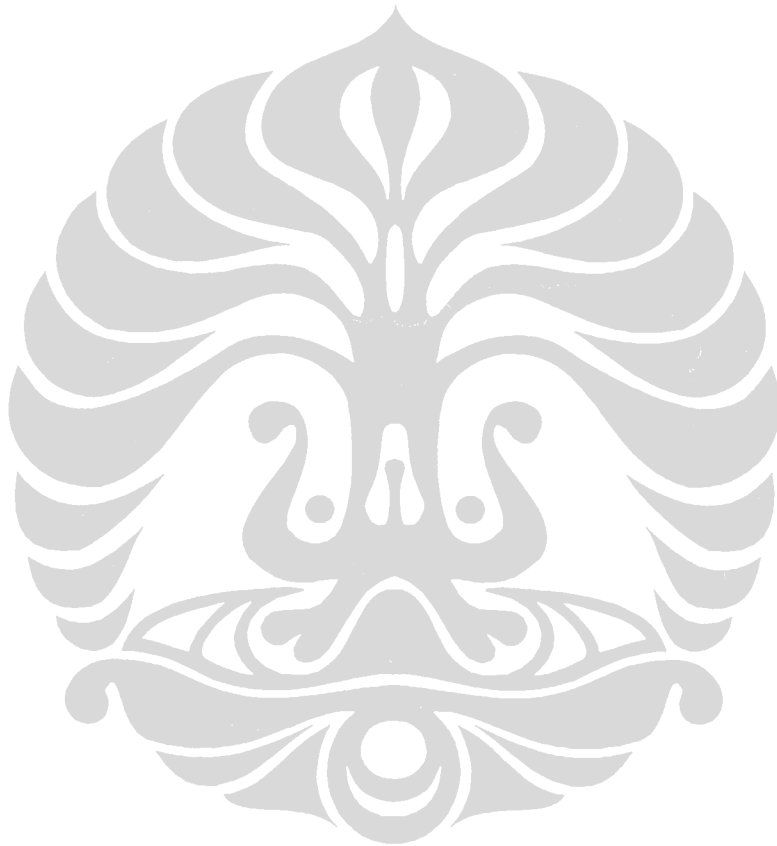
Gambar 3.7 Prosedur Pelayanan Paspor

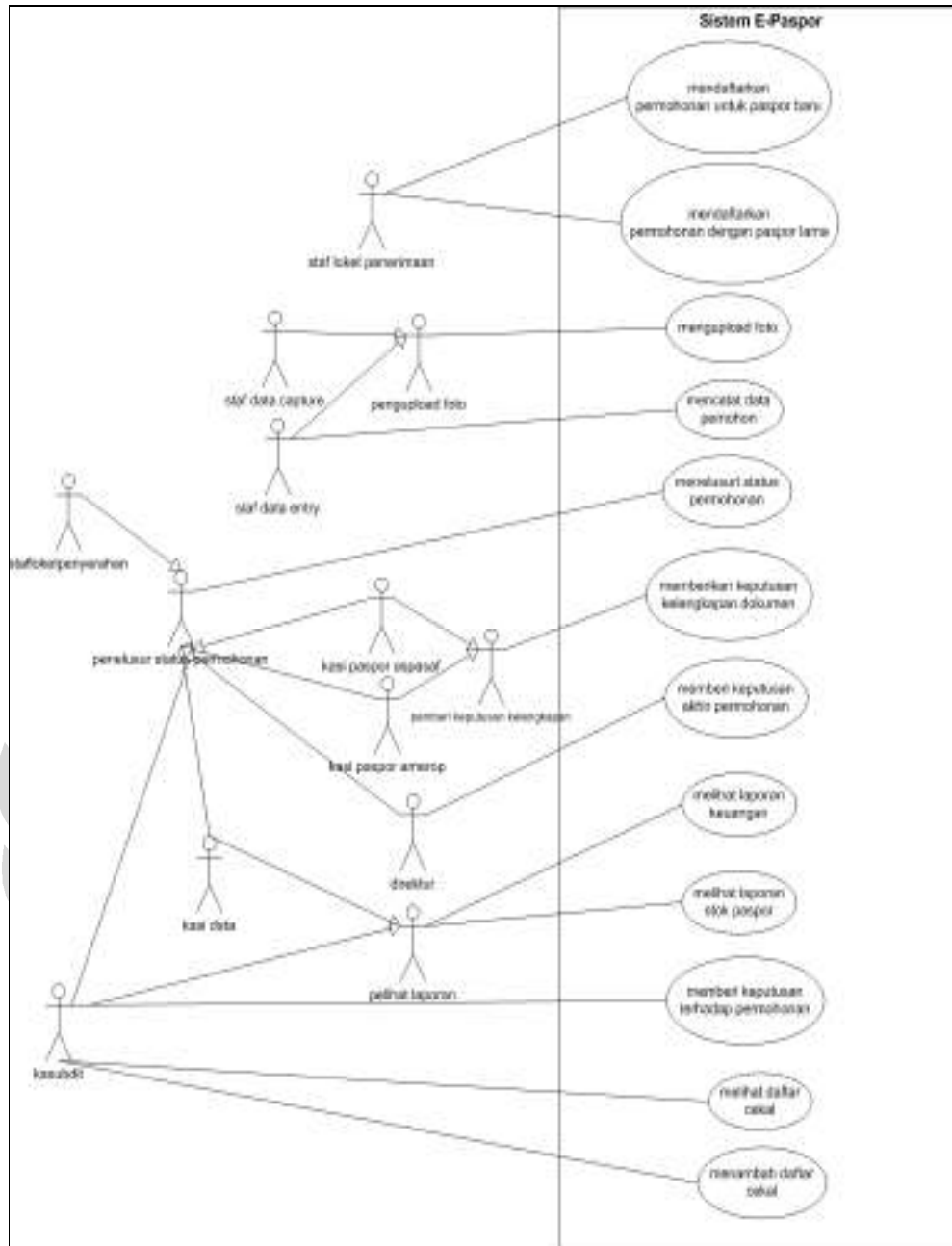
Gambar prosedur pelayanan paspor menunjukkan adanya aktivitas pelayanan paspor mulai dari staf loket penerimaan hingga staf loket penyerahan. Garis abu-abu yang utuh menandakan aktivitas yang dilakukan secara sistem E-Paspor dan garis abu-abu putus-putus menandakan aktivitas yang dilakukan secara manual. Sedangkan garis merah menunjukkan prosedur pelayanan paspor yang ditangani oleh E-Paspor dan yang akan diuji dengan Rational Robot dan Rational TestManager.

Dalam eksperimen ini, tidak ada dokumentasi *use case* yang lengkap yang dapat membantu proses penentuan *use case* apa saja yang akan diuji. Oleh karena itu, dilakukan *reverse engineering* untuk mendapatkan *use case* apa saja yang terdapat

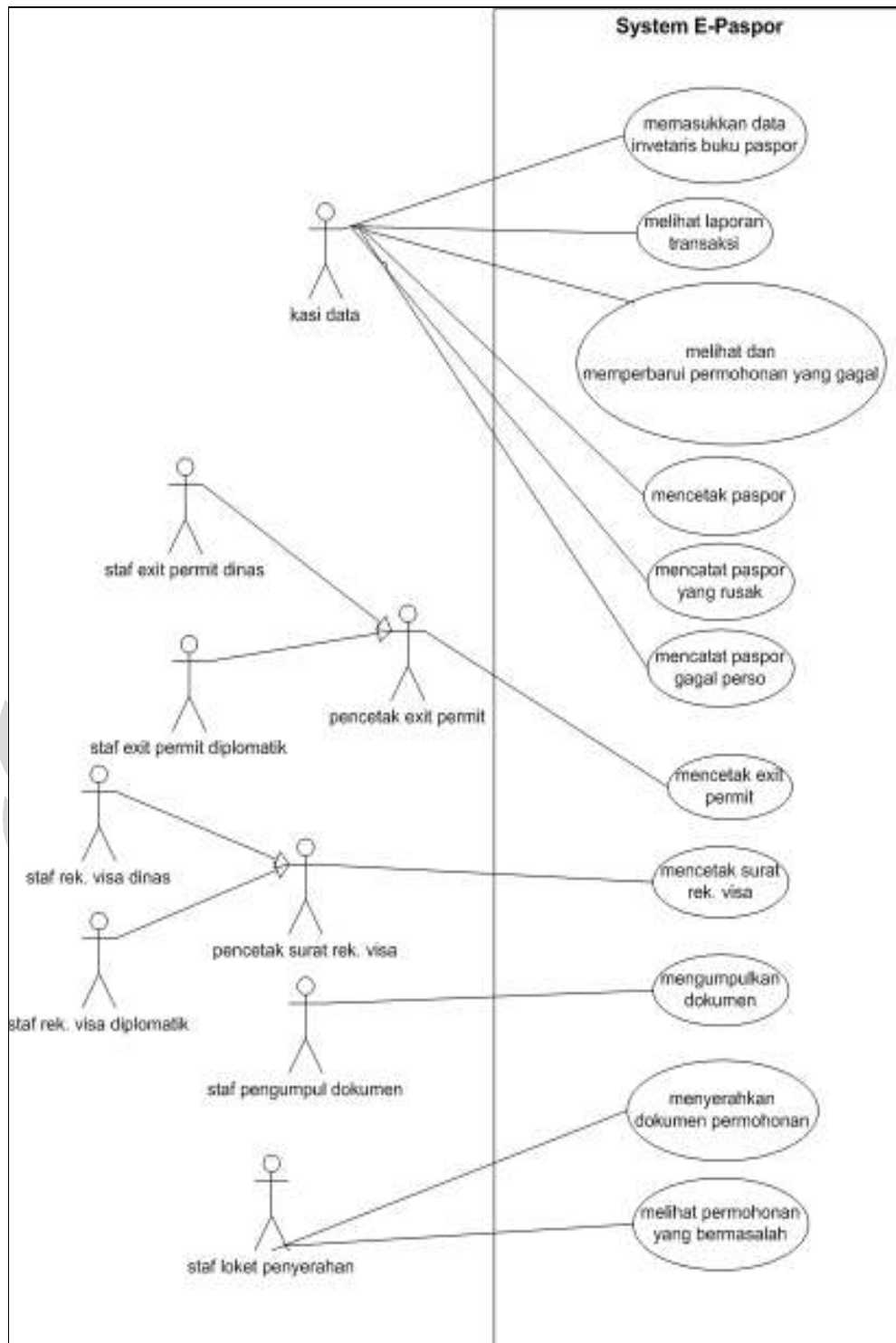
dalam aplikasi E-Paspor. Kegiatan ini dilakukan dengan terlebih dahulu mempelajari aplikasi E-Paspor dari dokumentasi *user manual*.

Hasil *reverse engineering* pada aplikasi E-Paspor tersebut adalah berupa diagram *use case* seperti yang diperlihatkan pada gambar *use case* yang terpisah menjadi dua berikut:





Gambar 3.8 Use Case E-Paspor – 1



Gambar 3.9 Use Case E-Paspor – 2 (sambungan)

Aplikasi E-Paspor digunakan oleh beberapa *user* atau pengguna, yaitu:

1. Administrator

2. Staf Loker Penerimaan
3. Staf Data Capture
4. Staf Data Entry
5. KASI Paspor AMEROP
6. KASI Paspor ASPASAF
7. Kasubdit
8. Direktur
9. KaSi Data
10. Staf Exit Permit Paspor Dinas
11. Staf Exit Permit Paspor Diplomatik
12. Staf Rekomendasi Visa Paspor Dinas
13. Staf Rekomendasi Visa Paspor Diplomatik
14. Staf Pengumpul Dokumen
15. Staf Loker Penyerahan

Berikut ini penjelasan untuk masing-masing *use case* aplikasi E-Paspor:

1. Mendaftarkan permohonan untuk paspor baru
Use case ini dijalankan apabila pemohon yang datang belum memiliki nomor e-paspor atau dengan kata lain, dirinya belum pernah menyampaikan permohonan e-paspor. Aktor *use case* ini adalah staf loket penerimaan yang pertama kali menerima permohonan dari pemohon. Staf loket penerimaan data memasukkan nama pemohon dan nomor identitasnya, serta jenis permohonan apa saja yang disampaikan, misalnya permohonan paspor baru, exit permit, dan atau surat rekomendasi visa.
2. Mendaftarkan permohonan untuk paspor lama
Aktor untuk *use case* ini juga adalah staf loket penerimaan. Pada *use case* ini, pemohon yang sudah pernah mendapatkan nomor e-paspor dapat mengajukan permohonan kembali untuk paspor baru, exit permit, dan atau surat rekomendasi visa.
3. Mengupload foto
Sebagai aktor yang menjalankan *use case* ini adalah staf data *capture* atau staf data *entry*. Salah satu dari kedua aktor tersebut dapat menjalankan *use*

case ini. Fungsi utama *use case* mengupload foto adalah untuk memasukkan foto yang telah di-*scan* dan atau diambil melalui kamera digital ke dalam profil pemohon untuk kemudian dicetak di paspor.

4. Mencatat data pribadi pemohon

Aktor yang menjalankan *use case* ini adalah staf data *entry*. Dengan *use case* ini, aktor dapat memasukkan data-data pribadi pemohon seperti tempat dan tanggal lahir, alamat rumah, alamat kantor, dan lain sebagainya. Dalam proses ini juga terdapat pencatatan negara mana yang akan dituju oleh pemohon, bulan keberangkatan, jenis visa, kawasan negara tujuan, dan lain-lain.

5. Menelusuri status permohonan

Use case ini dijalankan oleh aktor kasi paspor amerop (amerika dan eropa), kasi paspor aspasaf (asia, pasifik, dan afrika), kasubdit, direktur, kasi data, dan staf loket penyerahan. Fungsi utama dari *use case* ini adalah untuk melihat tahap-tahap pemeriksaan yang telah dilalui oleh suatu permohonan.

6. Memberikan keputusan kelengkapan dokumen

Sebagai aktor yang menjalankan *use case* ini adalah kasi paspor amerop dan atau kasi paspor aspasaf. Kasi paspor amerop akan menjalankan *use case* apabila negara tujuan suatu permohonan termasuk dalam kawasan amerika dan atau eropa. Sedangkan kasi paspor aspasaf akan menjalankan *use case* ini jika permohonan yang diajukan untuk mengunjungi negara yang termasuk dalam kawasan asia, pasifik, dan atau afrika. Fungsi utama *use case* ini adalah untuk memberikan keputusan apakah suatu permohonan disertai dengan dokumen-dokumen yang lengkap.

7. Memberikan keputusan terhadap permohonan

Aktor yang menjalankan *use case* ini adalah Kasubdit. Fungsi utama dari *use case* ini adalah untuk memberikan keputusan terhadap suatu permohonan, apakah disetujui, ditolak, ditunda, atau diserahkan kepada direktur.

8. Melihat laporan keuangan

Fungsi utama untuk *use case* ini adalah untuk melihat laporan keuangan berdasarkan jumlah dan jenis permohonan yang telah masuk. Aktor untuk *use case* ini adalah Kasubdit dan Kasi Data.

9. Melihat laporan stok paspor

Aktor yang menjalankan *use case* ini adalah Kasubdit dan Kasi Data. Dalam *use case* ini, aktor dapat melihat berapa jumlah stok paspor yang masih dimiliki dalam suatu periode.

10. Melihat daftar cekal

Use case ini digunakan oleh aktor, yaitu Kasubdit, untuk melihat daftar orang yang dicekal karena berbagai alasan agar tidak mendapatkan izin untuk ke luar negeri dengan paspor, exit permit, maupun surat rekomendasi visa.

11. Menambah daftar cekal

Fungsi utama *use case* ini adalah untuk memasukkan nama dan identitas orang-orang yang dicekal agar tidak mendapatkan izin ke luar negeri. Aktor yang menjalankan *use case* ini adalah Kasubdit.

12. Memberi keputusan akhir permohonan

Use case ini memungkinkan aktornya, yaitu Direktur, untuk memberikan keputusan terhadap suatu permohonan apabila di tahap sebelumnya, yakni oleh Kasubdit, keputusan terhadap permohonan tersebut diserahkan kepada Direktur. Direktur juga dapat memberikan keputusan terhadap permohonan yang belum ditangani oleh Kasubdit.

13. Memasukkan data inventaris buku paspor

Aktor untuk *use case* ini adalah Kasi Data. Dengan *use case* ini, aktor dapat memasukkan jumlah paspor baru (yang masih kosong) yang datang beserta rentang nomor paspornya.

14. Melihat laporan transaksi

Use case ini digunakan untuk melihat transaksi yang dilakukan pada bulan dan tahun tertentu. *Use case* ini dijalankan oleh aktor Kasi Data.

15. Melihat dan memperbarui permohonan yang gagal

Aktor untuk *use case* ini adalah Kasi Data. Aktor dapat melihat permohonan yang gagal dan dapat memprosesnya kembali.

16. Mencetak paspor

Fungsi utama dari *use case* ini adalah untuk mencetak paspor yang telah melalui langkah-langkah pemeriksaan dokumen. *Use case* ini dijalankan oleh aktor Kasi Data.

17. Mencatat paspor yang rusak

Dengan adanya *use case* ini, aktor, yaitu Kasi Data, dapat memasukkan nomor paspor yang buku paspornya rusak.

18. Mencatat paspor yang gagal di-perso

Aktor untuk *use case* ini adalah Kasi Data. Dengan *use case* ini, aktor dapat memasukkan nomor paspor yang gagal di-perso.

19. Mencetak exit permit

Aktor yang menjalankan *use case* ini adalah Staf Exit Permit Dinas dan atau Staf Exit Permit Diplomatik. Fungsi utama dari *use case* ini adalah untuk mencetak exit permit yang paspornya telah dicetak.

20. Mencetak surat rekomendasi visa

Fungsi utama dari *use case* ini sesuai dengan nama *use case*-nya, yakni untuk mencetak surat rekomendasi visa yang telah disetujui. Aktor yang menjalankan *use case* ini adalah Staf Rekomendasi Visa Dinas dan atau Staf Rekomendasi Visa Diplomatik.

21. Mengumpulkan dokumen

Aktor yang menjalankan *use case* ini adalah Staf Pengumpul Dokumen yang bertugas mengumpulkan dokumen-dokumen yang sudah tersedia, misalnya paspor, exit permit, dan rekomendasi visa yang sudah dicetak. Dengan *use case* ini, aktor dapat menandai sistem untuk dokumen-dokumen yang sudah terkumpul. Misalkan dari tiga permohonan (paspor baru, exit permit, dan surat rekomendasi visa), yang sudah terkumpul adalah paspor baru dan surat rekomendasi visa, maka yang ditandai di sistem adalah dokumen paspor baru dan surat rekomendas visa. Proses ini dilakukan untuk memberikan *flag* pada sistem bahwa dokumen-dokumen yang telah terkumpul siap diserahkan kepada pemohon.

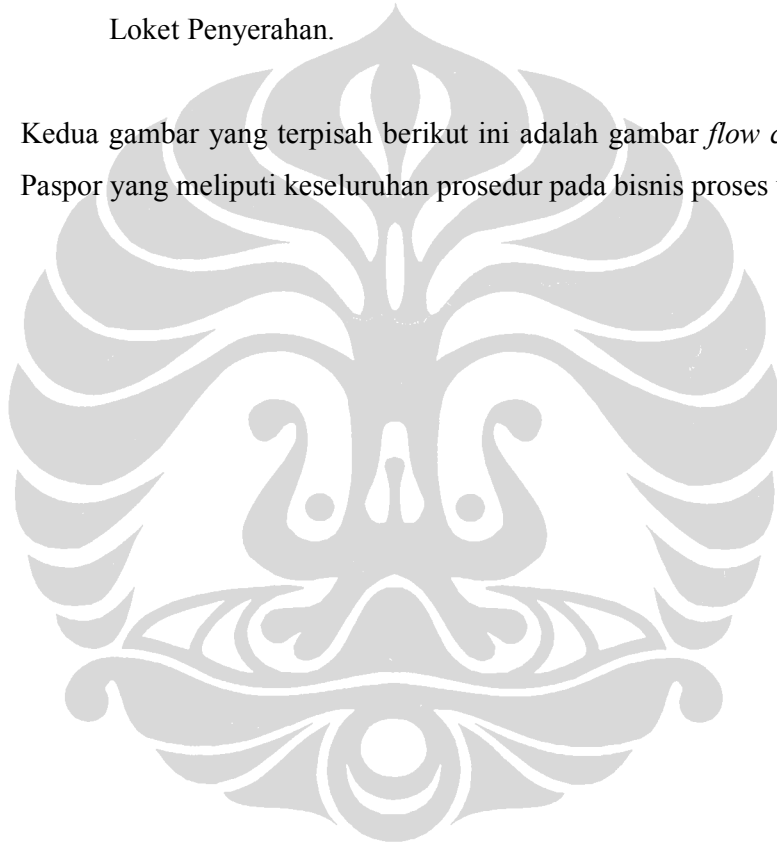
22. Menyerahkan dokumen-dokumen permohonan

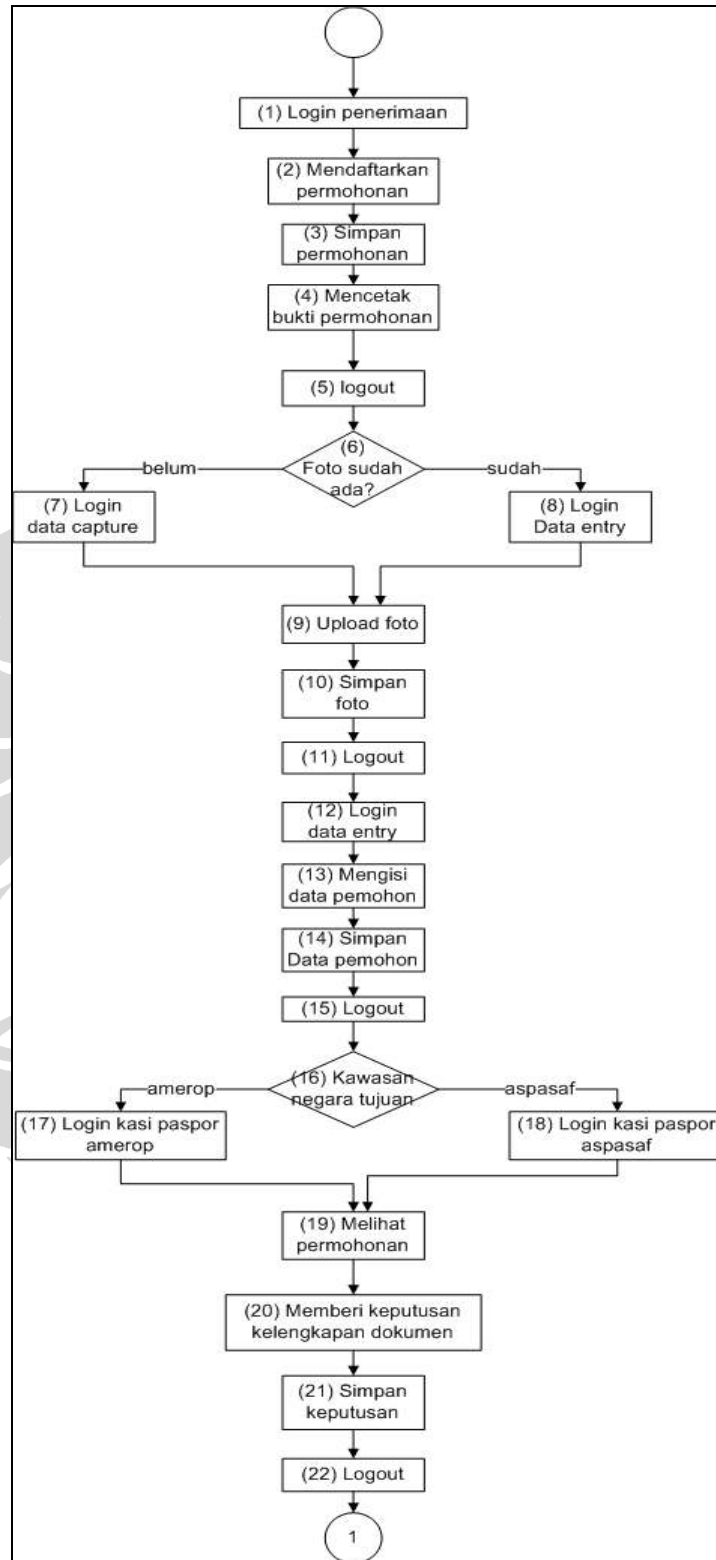
Aktor yang menggerakkan *use case* ini adalah Staf Loker Penyerahan. Dengan adanya *use case* ini, aktor dapat menandai dokumen apa saja yang diserahkan kepada pemohon sehingga sistem tidak menyimpan kembali permohonan yang dokumennya sudah diserahkan.

23. Melihat permohonan yang bermasalah

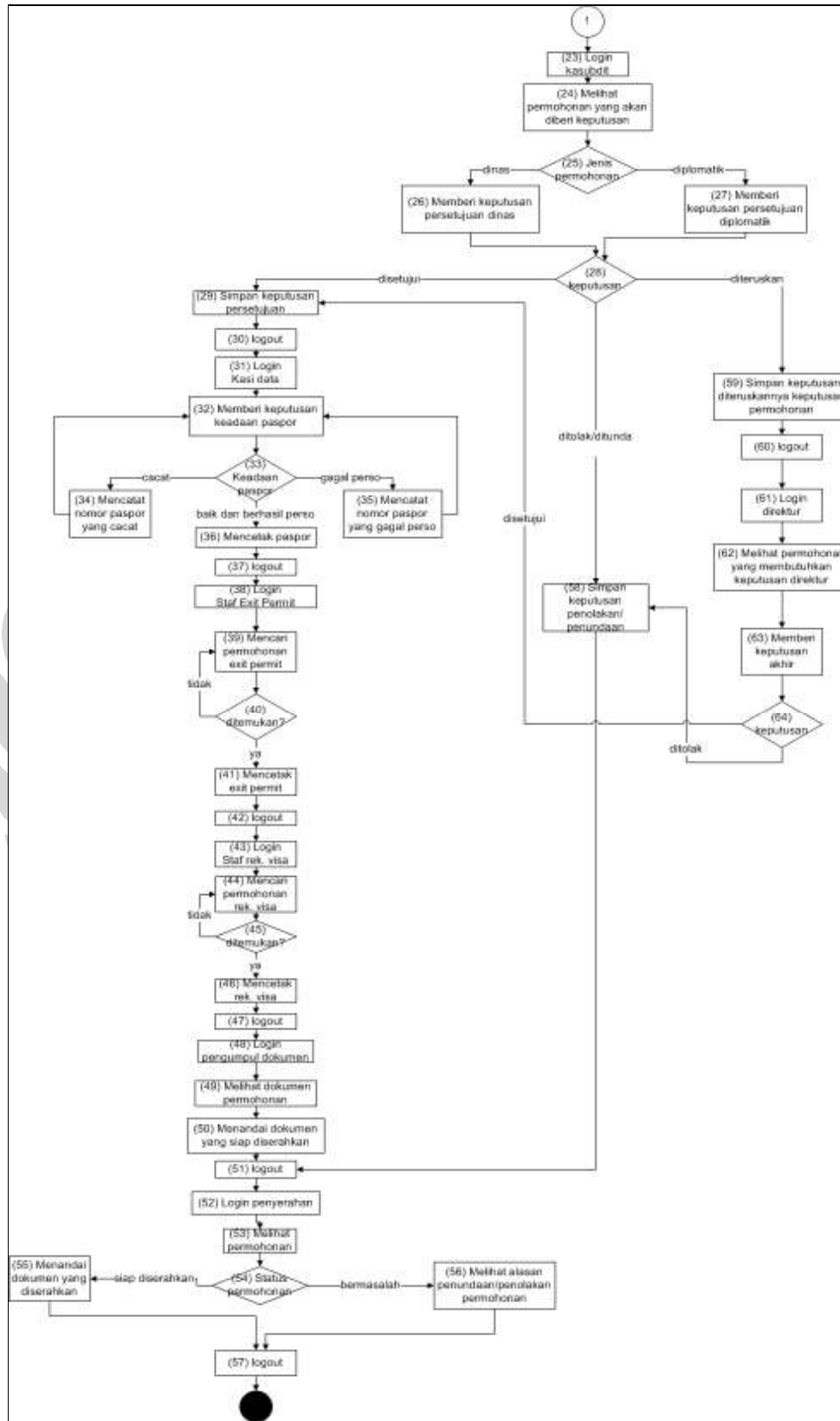
Fungsi utama *use case* ini adalah untuk melihat permohonan yang tidak dapat diserahkan (terdapat masalah sehingga tidak dapat diserahkan) kepada pemohon. Aktor yang dapat menjalankan *use case* ini adalah Staf Loker Penyerahan.

Kedua gambar yang terpisah berikut ini adalah gambar *flow chart* dari sistem E-Paspor yang meliputi keseluruhan prosedur pada bisnis proses utama.





Gambar 3.10 *Flow Chart* pada Proses Bisnis Utama Aplikasi E-Paspor -1



Gambar 3.11 Flow Chart pada Proses Bisnis Utama Aplikasi E-Paspor -2 (sambungan)
Universitas Indonesia

Dari gambar *flow chart* di atas, jumlah kemungkinan *path* yang bisa terjadi dalam alur proses E-Paspor dari awal pelayanan paspor hingga akhir dapat dihitung dengan melihat percabangan dari setiap bentuk belah ketupat (*decision*). Penghitungannya dijabarkan sebagai berikut:

$$\begin{aligned}
 \text{Jumlah Path Maksimum} &= \text{jumlah percabangan pada nomor 6 (2) x} \\
 &\quad \text{jumlah percabangan pada nomor 16 (2) x} \\
 &\quad \text{jumlah percabangan pada nomor 25 (2) x} \\
 &\quad ((\text{jumlah percabangan pada nomor 33 (3) x} \\
 &\quad \text{jumlah percabangan pada nomor 40 (2) x} \\
 &\quad \text{jumlah percabangan pada nomor 45 (2) x} \\
 &\quad \text{jumlah percabangan pada nomor 54 (1)) +} \\
 &\quad (\text{jumlah percabangan pada nomor 54 (1)) +} \\
 &\quad (\text{jumlah percabangan pada nomor 54 (1) +} \\
 &\quad (\text{jumlah percabangan pada nomor 33 (3) x} \\
 &\quad \text{jumlah percabangan pada nomor 40 (2) x} \\
 &\quad \text{jumlah percabangan pada nomor 45 (2) x} \\
 &\quad \text{jumlah percabangan pada nomor 54 (1)))) \\
 &= 2 \times 2 \times 2 \times ((3 \times 2 \times 2 \times 1) + (1) + (1 + (3 \times 2 \times 2 \times 1))) \\
 &= 8 \times (12 + 1 + 13) \\
 &= 208 \text{ path} \qquad (3,1)
 \end{aligned}$$

Jumlah percabangan pada nomor 54 dihitung hanya satu kemungkinan *path* yang muncul darinya. Hal ini dipengaruhi oleh *path* yang diambil pada percabangan sebelumnya. Misalkan pada percabangan pada nomor 28, *path* yang diambil adalah *path* “disetujui”, maka percabangan pada nomor 54 hanya ada satu kemungkinan *path*, yaitu *path* “siap diserahkan”. Contoh lain adalah pada percabangan nomor 28 apabila diambil *path* “ditolak/ditunda”, maka pada percabangan nomor 54, hanya akan ada satu kemungkinan *path* yang diambil, yaitu “bermasalah”. Berdasarkan penghitungan jumlah kemungkinan *path* maksimum, maka secara keseluruhan jumlah kemungkinan *path* untuk proses bisnis utama aplikasi E-Paspor adalah sejumlah 208 kemungkinan *path*.

Pada penghitungan tanpa menggunakan analisis terhadap ketergantungan antar percabangan, dihasilkan jumlah kemungkinan *path* sebanyak 416. Setelah dilakukan analisis seperti yang dilakukan pada penjelasan penghitungan di atas, kemudian jumlah tersebut tereduksi menjadi 208. Ke-208 kemungkinan *path* ini digunakan sebagai patokan jumlah *test case* yang harus dibuat untuk memenuhi pengujian yang menyeluruh. Jumlah ini cukup banyak untuk diimplementasikan dalam pengujian. Berdasarkan teori *cyclomatic complexity* oleh McCabe, analisis seberapa tinggi kerumitan sebuah sistem untuk diuji dapat dihitung dengan rumus sebagai berikut [PEA]:

$$V(G) = E - N + 2 \quad (3,2)$$

Dengan mensubstitusikan nilai E, yaitu jumlah *edge* (anak panah) yang berjumlah 77, dan nilai N, yaitu jumlah *node* (proses dan percabangan) yang berjumlah 64, maka didapatkan nilai *cyclomatic complexity* untuk *flowchart* E-Paspor di atas adalah sebesar 15. Berdasarkan penemuan yang dilakukan oleh Jones (1996), disimpulkan bahwa *cyclomatic complexity* 15 berarti bahwa sistem atau program cukup sulit dimengerti dan diuji [PEA]. Hal ini menyebabkan pengujian hanya dilakukan pada fungsi-fungsi tertentu saja. Pada pembahasan berikutnya akan dipilih beberapa *use case* yang akan direncanakan untuk diuji sehingga *test case* yang dihasilkan pun dapat diimplementasikan dengan mudah.

Dari ke-23 *use case* E-Paspor, hanya 11 *use case* di antaranya yang merupakan proses bisnis utama aplikasi. Kesebelas *use case* tersebut digunakan sebagai objek pengujian, yaitu:

1. Mendaftarkan permohonan untuk paspor baru oleh aktor Staf Loker Penerimaan.
2. Mendaftarkan permohonan untuk paspor lama oleh aktor Staf Loker Penerimaan.
3. Mengupload foto oleh Staf Data Capture dan atau Staf Data Entry.
4. Mencatat data pribadi pemohon oleh Staf Data Entry.

5. Memberikan keputusan kelengkapan dokumen oleh Kasi Paspor Amerop dan atau Kasi Paspor Aspasaf.
6. Memberikan keputusan terhadap permohonan oleh Kasubdit.
7. Mencetak paspor oleh Kasi Data.
8. Mencetak exit permit oleh Staf Exit Permit Dinas dan atau Staf Exit Permit Diplomatik.
9. Mencetak surat rekomendasi visa oleh Staf Rekomendasi Visa Dinas dan atau Staf Rekomendasi Visa Diplomatik.
10. Mengumpulkan dokumen-dokumen permohonan oleh Staf Pengumpul Dokumen.
11. Menyerahkan dokumen oleh Staf Loker Penyerahan.

Pembentukan *test case* dilakukan dengan terlebih dahulu melihat *basic flow* dan *alternate flow* dari masing-masing *use case*. Dikarenakan terbatasnya dokumentasi sistem, maka *basic flow* dan *alternate flow* juga dibuat secara *reverse engineering* seperti yang dilakukan dalam pembuatan *use case*. Dua dari kesebelas tabel menunjukkan *basic flow* beserta *alternate flow* sesuai dengan *use case*-nya masing-masing. Secara lengkap, keseluruhan tabel disajikan pada Lampiran 1.

Tabel 3.1 Basic Flow dan Alternate Flow untuk Use Case Mendaftarkan Permohonan untuk Paspor Baru

Nomor	1
Nama <i>use case</i>	Mendaftarkan permohonan untuk paspor baru
Aktor	Staf Loker Penerimaan
<i>Basic flow</i>	<ol style="list-style-type: none"> 1. login 2. sistem menampilkan halaman pendaftaran permohonan 3. pilih “Belum memiliki e-paspor” 4. masukkan nama, identitas pemohon, jenis permohonan, dan macam permohonan 5. <i>submit</i>/simpan permohonan 6. sistem menyimpan permohonan 7. cetak bukti permohonan

	8. dengan terhubung dengan <i>printer</i> , sistem mencetak bukti permohonan
<i>Alternate flow</i>	1. <i>username/password</i> salah 2. keluar aplikasi 3. data <i>mandatory</i> yang harus diisikan tidak lengkap

Pada tabel 3.1, *alternate flow* yang pertama mengacu pada *basic flow* yang pertama, yaitu apabila login gagal karena kesalahan *username* dan atau *password*. Sedangkan *alternate flow* yang ketiga mengacu pada *basic flow* yang keempat. Dalam hal ini data *mandatory* yang harus diisikan adalah nama dan identitas. Jika salah satu atau keduanya kosong atau tidak sesuai format, maka akan dianggap data *mandatory* tidak lengkap.

Tabel 3.2 Basic Flow dan Alternate Flow untuk Use Case Mendaftarkan Permohonan untuk Paspor Lama

Nomor	2
Nama <i>use case</i>	Mendaftarkan permohonan untuk paspor lama
Aktor	Staf Loker Penerimaan
<i>Basic flow</i>	1. login 2. sistem menampilkan halaman pendaftaran permohonan 3. pilih “Sudah memiliki e-paspor” 4. masukkan nomor e-paspor, jenis permohonan, dan macam permohonan 5. <i>submit</i> /simpan permohonan 6. sistem menyimpan permohonan 7. cetak bukti permohonan 8. dengan terhubung dengan <i>printer</i> , sistem mencetak bukti permohonan
<i>Alternate flow</i>	1. <i>username/password</i> salah 2. keluar aplikasi 3. nomor e-paspor tidak valid

Pada tabel 3.2, *alternate flow* yang pertama mengacu pada *basic flow* yang pertama, yaitu apabila login gagal karena kesalahan *username* dan atau *password*.

Sedangkan *alternate flow* yang ketiga mengacu pada *basic flow* yang keempat, yaitu apabila nomor e-paspor yang dimasukkan tidak valid.

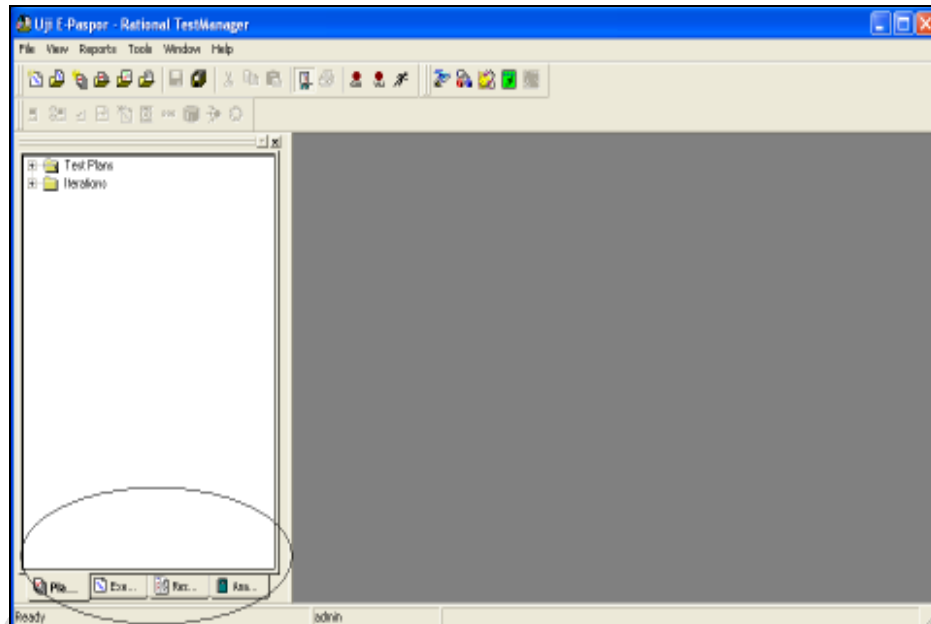
3.2.2 Pembentukan *test plan* dari *use case* yang telah ditentukan

Sebagai tahap perencanaan dalam proses pengujian perangkat lunak dengan menggunakan Rational TestManager, *test plan* merupakan hasil nyatanya. Untuk itu, *test plan* yang sudah ditentukan harus dimasukkan sebagai *test asset* ke dalam TestManager. *Test plan* bisa diambil langsung dari *use case* pada aplikasi yang akan diuji. Karena terdapat 11 *use case* yang diuji, maka berikut ini 11 daftar *test plan* yang sama dengan *use case*-nya yang dihasilkan dari tahap perencanaan pengujian:

1. Mendaftarkan permohonan untuk paspor baru
2. Mendaftarkan permohonan untuk paspor lama
3. Mengupload foto
4. Mencatat data pribadi pemohon.
5. Memberikan keputusan kelengkapan dokumen.
6. Memberikan keputusan terhadap permohonan.
7. Mencetak paspor.
8. Mencetak exit permit.
9. Mencetak surat rekomendasi visa.
10. Mengumpulkan dokumen-dokumen permohonan.
11. Menyerahkan dokumen.

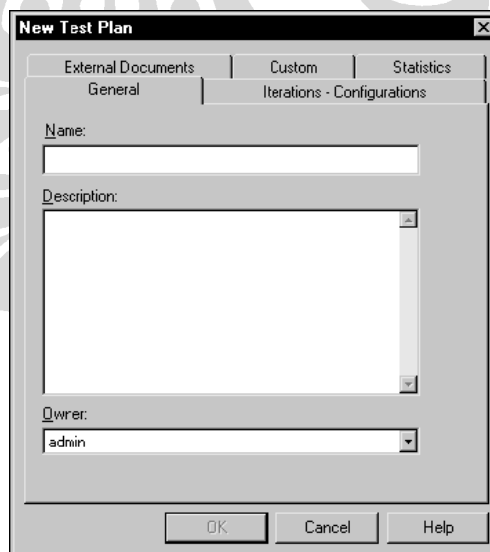
Kesebelas *test plan* di atas kemudian dicatat dalam Rational TestManager dengan cara sebagai berikut:

1. Buka Rational TestManager dari menu Start
2. Muncul *window* Rational Test Login → pilih nama *project*, dalam hal ini Uji E-Paspor → OK
3. Pilih *View*, pastikan Test Asset Workspace tercentang untuk menampilkan Test Asset Workspace yang ditampilkan di *window* TestManager sebelah kiri yang terdiri dari tab *Planning*, *Execution*, *Result*, dan *Analysis* seperti yang terlihat pada gambar berikut:



Gambar 3.12 Tampilan Test Asset Workspace

4. Pilih tab *Planning* → klik kanan pada *folder* Test Plans → pilih New Test Plan maka akan ditampilkan *window* seperti berikut ini.

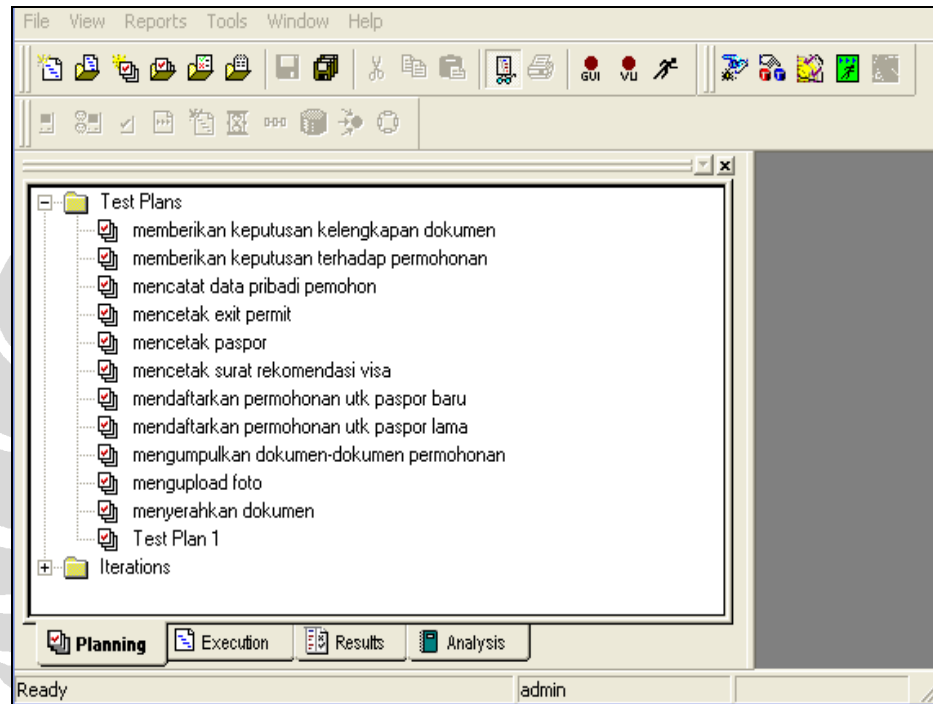


Gambar 3.13 Membuat *Test Plan* Baru

5. Masukkan nama *test plan* dan deskripsinya sesuai dengan nama-nama *test plan* yang telah disebutkan sebelumnya. Pada tab *External Documents*

dapat dimasukkan misalkan dokumen *Microsoft Word* yang memiliki deskripsi lebih jelas mengenai *test plan* yang bersangkutan. Dalam hal ini, tidak ada dokumen yang terkait.

6. Ulangi langkah 4 dan 5 hingga semua *test plan* tercatat dalam Rational TestManager seperti yang terlihat pada gambar berikut:



Gambar 3.14 Semua *Test Plan* Tercatat dalam TestManager

3.2.3 Pembentukan *test case* dari *use case* yang telah ditentukan

Dengan mengacu pada tutorial *Generating Test Cases from Use Cases* oleh Jim Heumann, pembuatan *test case* kini sudah dapat dimulai karena telah diketahui *basic flow* dan *alternate flow* dari masing-masing *use case* yang telah ditentukan untuk diuji. Tahapan pembentukan *test case* terdiri dari:

1. Pembentukan skenario pengujian
2. Pembentukan *test case*
3. Penggunaan nilai data dalam *test case*

3.2.3.1 Pembentukan skenario pengujian

Skenario pengujian dapat dibentuk dengan mengkombinasikan *basic flow* dan *alternate flow*. Dengan mengacu pada *basic flow* dan *alternate flow* yang telah dibentuk untuk masing-masing *use case* yang terlihat pada Lampiran 1, subbab ini akan menjabarkan cara untuk membentuk skenario pengujian. Tabel-tabel berikut menunjukkan hasil kombinasi *basic flow* dan *alternate flow* untuk masing-masing *use case* yang masing-masing kombinasi menjadi sebuah skenario.

Tabel 3.3 Daftar Skenario dari Use Case Mendaftarkan Permohonan untuk Paspor Baru

<i>Use case</i> Mendaftarkan Permohonan untuk Paspor Baru		
Skenario 1	<i>Basic flow</i>	
Skenario 2	<i>Basic flow</i>	<i>Alternate flow</i> 1 – <i>username/password</i> salah
Skenario 3	<i>Basic flow</i>	<i>Alternate flow</i> 2 – keluar sistem
Skenario 4	<i>Basic flow</i>	<i>Alternate flow</i> 3 – isian tidak lengkap

Dengan merujuk pada tabel 3.1, tabel 3.3 menunjukkan hasil kombinasi *basic flow* dan *alternate flow* yang menghasilkan 4 skenario pengujian. Skenario yang pertama sesuai dengan *basic flow*, yang kedua merupakan kombinasi antara *basic flow* dengan *alternate flow* pertama, skenario tiga merupakan kombinasi *basic flow* dengan *alternate flow* yang kedua, dan skenario keempat merupakan kombinasi antara *basic flow* dan *alternate flow* yang ketiga.

Tabel 3.4 Daftar Skenario dari Use Case Mendaftarkan Permohonan untuk Paspor Lama

<i>Use case</i> Mendaftarkan Permohonan untuk Paspor Lama		
Skenario 5	<i>Basic flow</i>	
Skenario 6	<i>Basic flow</i>	<i>Alternate flow</i> 1 – <i>username/password</i> salah
Skenario 7	<i>Basic flow</i>	<i>Alternate flow</i> 2 – keluar sistem
Skenario 8	<i>Basic flow</i>	<i>Alternate flow</i> 3 – nomor e-paspor tidak valid

Dengan merujuk pada tabel 3.2, tabel 3.4 menunjukkan hasil kombinasi *basic flow* dan *alternate flow* yang menghasilkan 4 skenario pengujian. Skenario yang pertama sesuai dengan *basic flow*, yang kedua merupakan kombinasi antara *basic*

flow dengan *alternate flow* pertama, skenario tiga merupakan kombinasi *basic flow* dengan *alternate flow* yang kedua, dan skenario keempat merupakan kombinasi antara *basic flow* dan *alternate flow* yang ketiga.

Kedua tabel daftar skenario tersebut merupakan dua contoh cara membuat skenario. Untuk lebih lengkapnya, seluruh tabel daftar skenario disajikan pada Lampiran 2.

Dengan demikian, jumlah skenario yang berhasil dibentuk adalah sejumlah 29 skenario yang dapat dilihat pada Lampiran 2.. Namun, dikarenakan keterbatasan waktu pengujian, maka skenario pengujian yang digunakan adalah skenario yang berasal dari *basic flow* dari masing-masing *use case*. Jadi, skenario yang akan digunakan untuk pengujian adalah sejumlah 11 skenario karena masing-masing *use case* memiliki satu buah skenario yang berasal dari *basic flow* saja.

3.2.3.2 Pembentukan *test case* awal

Setelah dibentuknya skenario pengujian, langkah selanjutnya dalam proses pembentukan *test case* adalah pembentukan *test case* awal. Langkah ini dilakukan dengan melihat skenario yang telah dibentuk. Perlu dijelaskan kembali, bahwa skenario yang digunakan untuk pengujian adalah skenario yang berasal dari *basic flow* dari masing-masing *use case* sehingga jumlah skenario adalah 11 buah. Paling tidak, satu skenario menghasilkan satu *test case*. Proses penentuan *test case* ditunjukkan pada tabel-tabel berikut.

Tabel 3.5 Daftar *Test Case* dari Skenario 1

Skenario 1 dari <i>Use case</i> Mendaftarkan permohonan untuk paspor baru									
<i>Test case ID</i>	Skenario/ kondisi	<i>Username</i>	<i>Password</i>	Isian data lengkap	Jenis Permohonan	Permohonan Paspor Baru	Permohonan Exit Permit	Permohonan Surat Rek. Visa	<i>Expected Result</i>
1	Pendaftaran untuk paspor baru sukses	V	V	V	V	V	N/A	N/A	Muncul nomor registrasi
2	Pendaftaran untuk paspor baru dan exit permit sukses	V	V	V	V	V	V	N/A	Muncul nomor registrasi
3	Pendaftaran untuk paspor baru, exit permit, dan rek. visa sukses	V	V	V	V	V	V	V	Muncul nomor registrasi

Tabel 3.6 Daftar *Test Case* dari Skenario 2

Skenario 2 dari <i>use case</i> Mendaftarkan permohonan untuk paspor lama								
<i>Test case ID</i>	Skenario/ kondisi	<i>Username</i>	<i>Password</i>	Nomor e-paspor	Permohonan Paspor Baru	Permohonan Exit Permit	Permohonan Surat Rek. Visa	<i>Expected Result</i>
4	Pendaftaran dengan paspor lama untuk permohonan exit permit saja	V	V	V	N/A	V	N/A	Muncul nomor registrasi

Tabel 3.5 menunjukkan daftar *test case* yang dihasilkan dari skenario 1, yaitu pendaftaran untuk paspor baru berhasil dilakukan. Pendaftaran untuk paspor baru ini terdapat tiga macam permohonan, yaitu paspor baru, exit permit, dan surat rekomendasi visa. Masing-masing *test case* pada tabel 3.5 mewakili permohonan paspor baru saja, permohonan paspor baru dan exit permit, dan permohonan paspor baru, exit permit, dan rekomendasi visa.

Tabel 3.6 menunjukkan daftar *test case* yang dihasilkan dari skenario 2, yaitu pendaftaran dengan paspor lama berhasil dilakukan. Pendaftaran dengan menggunakan paspor lama ini juga terdapat tiga macam permohonan, yaitu paspor baru, exit permit, dan surat rekomendasi visa. Akan tetapi, *test case* yang dibuat hanya untuk pengujian pada permohonan exit permit saja seperti yang terlihat pada tabel 3.6 tersebut. Hal ini dikarenakan adanya keterbatasan waktu pengujian.

Tabel-tabel daftar *test case* yang dibuat berdasarkan skenario-skenario dalam *use case* secara lengkap disajikan pada Lampiran 3. Kedua tabel daftar *test case* yang telah disajikan hanya merupakan dua contoh dalam membuat *test case*.

3.2.3.3 Penggunaan nilai data dalam *test case*

Diperlukan adanya nilai data yang di-*assign* ke dalam setiap *test case* untuk menggantikan nilai V (valid) dan I (invalid). Hal ini akan menyebabkan perubahan jumlah *test case* yang akan digunakan untuk pengujian. Tabel-tabel berikut menunjukkan penggantian nilai V menjadi nilai data (tidak ada nilai I karena skenario yang diambil sebagai *test case* adalah hanya yang *basic flow*).

Tabel 3.7 Test Case dari Skenario 1 dengan Nilai Data

Skenario 1 dari Use case Mendaftarkan permohonan untuk paspor baru									
Test case ID	Skenario/kondisi	Username	Password	Isian data lengkap	Jenis Permohonan	Permohonan Paspor Baru	Permohonan Exit Permit	Permohonan Surat Rek. Visa	Expected Result
1	Pendaftaran untuk paspor baru sukses	stafloketpenerimaan	stafloketpenerimaan	V	Dinas, Diplomatik	V	N/A	N/A	Muncul nomor registrasi
2	Pendaftaran untuk paspor baru dan exit permit sukses	stafloketpenerimaan	stafloketpenerimaan	V	Dinas, Diplomatik	V	V	N/A	Muncul nomor registrasi
3	Pendaftaran untuk paspor baru, exit permit, dan rek. visa sukses	stafloketpenerimaan	stafloketpenerimaan	V	Dinas, Diplomatik	V	V	V	Muncul nomor registrasi

Tabel 3.8 Test Case dari Skenario 2 dengan Nilai Data

Skenario 2 dari use case Mendaftarkan permohonan untuk paspor lama								
Test case ID	Skenario/kondisi	Username	Password	Nomor e-paspor	Permohonan Paspor Baru	Permohonan Exit Permit	Permohonan Surat Rek. Visa	Expected Result
4	Pendaftaran dengan paspor lama untuk permohonan exit permit saja	stafloketpenerimaan	stafloketpenerimaan	S000001	N/A	V	N/A	Muncul nomor registrasi

Berdasarkan tabel 3.7, terlihat adanya penggantian nilai V pada kolom *username*, *password*, dan jenis permohonan. Pada kolom-kolom yang lain tidak diperlukan adanya penggantian nilai.

Berdasarkan tabel 3.8, terlihat adanya penggantian nilai V pada kolom *username* *password*, dan nomor paspor. Pada kolom-kolom yang lain tidak diperlukan adanya penggantian nilai.

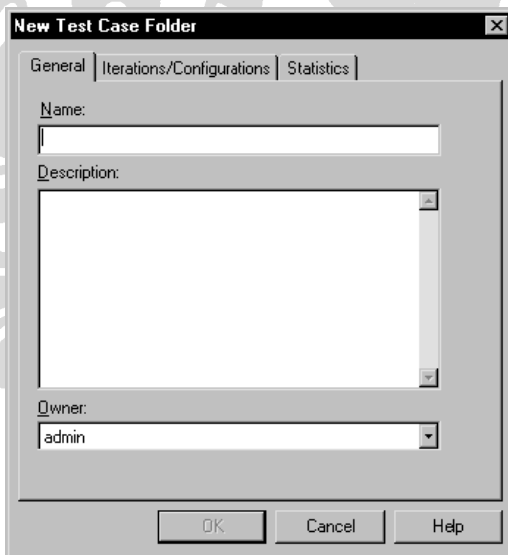
Kedua tabel yang telah ditampilkan hanyalah dua buah contoh hasil penggantian nilai V dan I pada pembuatan *test case*. Tabel-tabel lain secara lengkap disajikan pada Lampiran 4. Ringkasnya, *test case* yang dihasilkan dalam proses ini adalah sebagai berikut:

1. Sukses mendaftarkan permohonan paspor baru saja
2. Sukses mendaftarkan permohonan paspor baru dan exit permit
3. Sukses mendaftarkan permohonan paspor baru, exit permit, dan rek. visa
4. Sukses mendaftarkan untuk permohonan exit permit saja dengan menggunakan e-paspor lama
5. Sukses meng-upload foto
6. Sukses mencatat data pribadi pemohon
7. Sukses memberikan keputusan kelengkapan dokumen untuk jenis permohonan paspor baru
8. Sukses memberikan keputusan kelengkapan dokumen untuk jenis permohonan paspor baru dan exit permit
9. Sukses memberikan keputusan kelengkapan dokumen untuk jenis permohonan paspor baru, exit permit, dan rekomendasi visa
10. Sukses memberikan keputusan kelengkapan dokumen untuk jenis permohonan exit permit
11. Sukses memberikan keputusan terhadap jenis permohonan paspor saja
12. Sukses memberikan keputusan terhadap jenis permohonan paspor dan exit permit
13. Sukses memberikan keputusan terhadap jenis permohonan paspor, rekomendasi visa, dan exit permit

14. Sukses memberikan keputusan terhadap jenis permohonan exit permit
15. Sukses mencetak paspor
16. Sukses mencetak exit permit
17. Sukses mencetak surat rekomendasi visa
18. Sukses menandai dokumen yang diminta bahwa sudah tersedia dan siap diserahkan
19. Sukses menandai dokumen yang sudah diserahkan

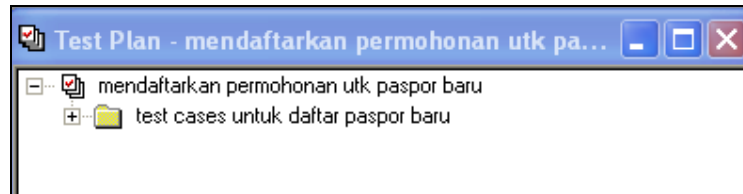
Setelah mendapatkan *test case* yang diperlukan untuk pengujian sebagai aset pengujian, maka tahap selanjutnya adalah memasukkan *test case* tersebut ke dalam Rational TestManager dengan cara sebagai berikut:

1. Pada *Test Asset Workspace*, pilih tab Planning
2. Pilih salah satu *test plan* yang telah dibuat sebelumnya, misalnya “Mendaftarkan permohonan utk paspor baru” → klik kanan → pilih Open
3. Pada *window Test Plan*, klik kanan pada *test plan* → pilih *Insert Test Case Folder*, maka akan tampil *window* sebagai berikut:



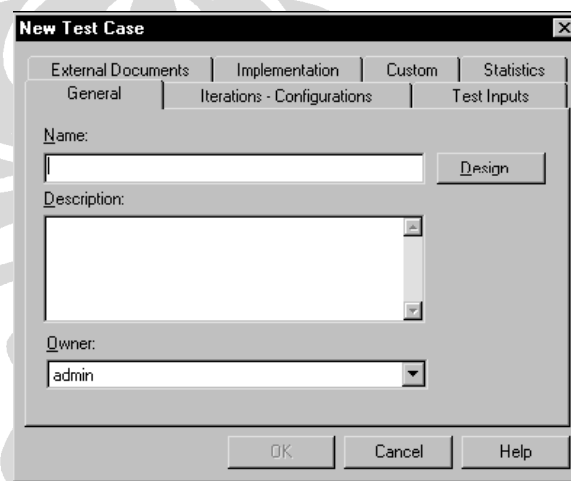
Gambar 3.15 *Window* untuk Membuat *Test Case Folder*

4. Masukkan nama (misalnya “*test cases* untuk daftar paspor baru”) dan deskripsi dari *test case folder* → OK, maka pada *window Test Plan* akan tampil:



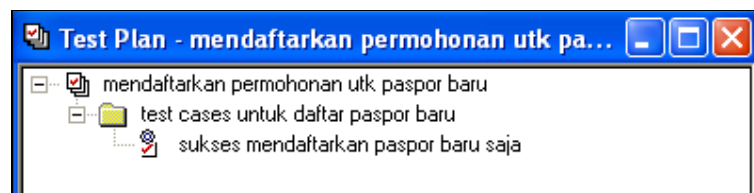
Gambar 3.16 Test Case Folder Berhasil Dibuat

5. Klik kanan pada folder *test case* → pilih *Insert Test Case*, maka akan muncul *window* isian properti dari *test case* sebagai berikut:



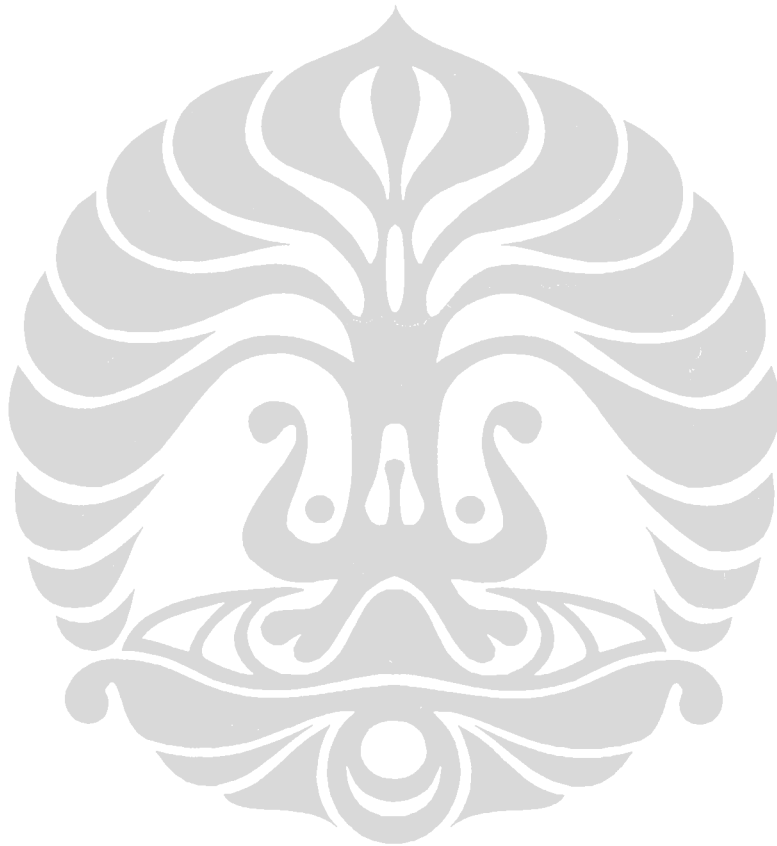
Gambar 3.17 Window Isian Properti Test Case

6. Masukkan nama *test case* (misalnya “sukses mendaftarkan paspor baru saja”) dan deskripsi *test case* → OK, kemudian nama *test case* yang sudah dimasukkan dapat dilihat pada *window Test Plan* sebagai berikut:



Gambar 3.18 Test Case Berhasil Dimasukkan ke Dalam Folder Test Case

7. Ulangi langkah 5 dan 6 untuk membuat *test case* baru pada folder yang sama. Ulangi langkah 2 hingga 6 untuk membuat *test case* baru pada folder dan *test plan* yang berbeda



BAB 4 IMPLEMENTASI

Bab ini menjelaskan bagaimana mengimplementasikan *test case* yang sudah didapatkan pada Bab 3 ke dalam *test script*. Pembuatan *test script* dilakukan dengan menggunakan Rational Robot yang dapat men-*generate test script* setelah melakukan perekaman terlebih dahulu.

4.1 *Record dan Playback Script*

Dalam uji fungsionalitas (*functional testing*), Rational Robot dapat merekam aksi yang dilakukan oleh *tester* terhadap aplikasi yang berbasis GUI, misalnya HTML, Java, Visual Basic, Delphi, dan lain-lain. Setelah selesai melakukan perekaman, Robot menghasilkan *script* yang kemudian dapat di-*playback* kembali sehingga mengulang semua aksi yang pernah dilakukan. Kali ini, dengan menggunakan studi kasus aplikasi E-Paspor, perekaman dilakukan terhadap objek-objek HTML pada tampilan GUI E-Paspor.

Berikut ini dijelaskan bagaimana cara melakukan perekaman dan pengulangan *script* pada aplikasi E-Paspor:

1. Pastikan bahwa *window* yang aktif hanya browser yang sudah menampilkan GUI E-Paspor dan Rational Robot. Misalkan GUI E-Paspor yang akan digunakan sebagai objek perekaman adalah GUI pendaftaran pemohon seperti yang terlihat pada gambar berikut.



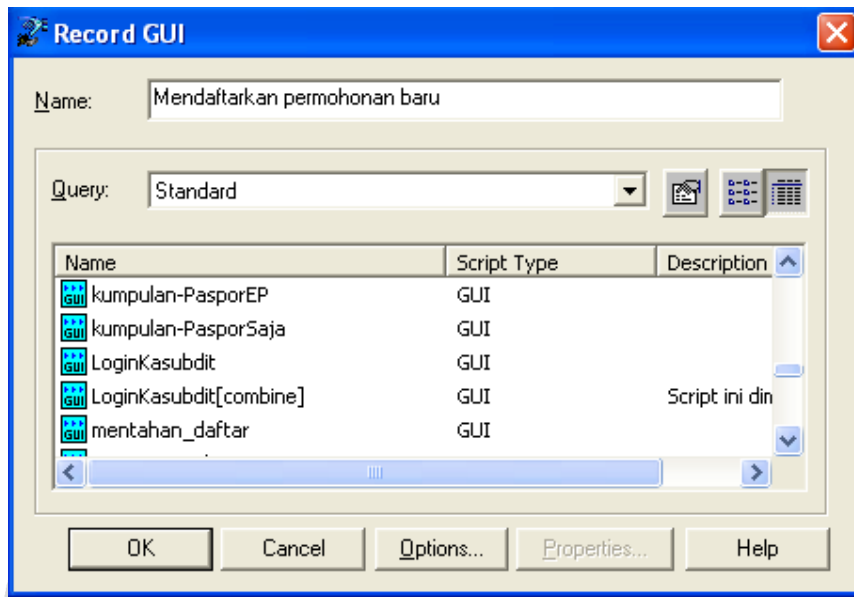
Gambar 4.1 Contoh Tampilan GUI E-Paspor

2. Pada Rational Robot, klik tombol GUI pada *toolbar* seperti pada gambar berikut:



Gambar 4.2 Tombol Perekaman GUI pada Rational Robot

3. Masukkan nama *script*, misalnya “mendaftarkan permohonan baru”, dan klik OK



Gambar 4.3 Memasukkan Nama Script

Perhatikan apa yang terjadi. Pada saat perekaman dimulai, Rational Robot me-*minimize*-kan *window*-nya sendiri agar aksi-aksi (misalnya klik) dapat dilakukan terhadap objek-objek dalam GUI E-Paspor. Sebuah *toolbar* perekaman GUI muncul menggantikan *window* Rational Robot.



Gambar 4.4 Toolbar Perekaman GUI

Toolbar tersebut akan selalu muncul di atas *window-window* yang lain. Hal ini menunjukkan bahwa Rational Robot sedang dalam keadaan merekam. Rational Robot merekam semua interaksi yang dilakukan terhadap objek-objek apapun pada *window* yang sedang terbuka. Apabila terdapat sesuatu yang harus dilakukan di luar interaksi dengan *window* E-Paspor, gunakan tombol *Pause* pada *toolbar* perekaman GUI.



Gambar 4.5 Tombol *Pause* pada *Toolbar* Perekaman GUI

Dengan menekan tombol *Pause*, Rational Robot akan mengabaikan semua aksi yang dilakukan selama perekaman dihentikan (*pause*). Proses perekaman dapat dilanjutkan kembali dengan menekan tombol *Pause* lagi.

4. Lakukan aksi apapun pada *window* E-Paspor, misalnya klik pada *editbox* Nama Lengkap, ketikkan “Rina Violyta”, tab, ketikkan “12500080Y”, dan pilih “Diplomatik” pada *combolistbox*
5. Setelah selesai melakukan interaksi dengan GUI E-Paspor, klik tombol *Stop* pada *toolbar* perekaman GUI



Gambar 4.6 Tombol *Stop* pada *Toolbar* Perekaman GUI

Dengan demikian, berakhirlah proses perekaman interaksi pada GUI E-Paspor. Hasil perekaman ini adalah sebuah *script* yang berisi kode-kode SQABasic yang merupakan penerjemahan dari setiap interaksi menjadi kode program. *Script* yang dihasilkan untuk interaksi yang dilakukan pada langkah 4 adalah sebagai berikut:

Script 1

```

1 Sub Main
2   Dim Result As Integer
3
4   'Initially Recorded: 6/10/2009 10:47:49 AM
5   'Script Name: Mendaftarkan permohonan baru
6
7   Window SetContext, "Caption=Sistem Pelayanan ePaspor Departemen Luar Negeri RI -
Microsoft Internet Explorer", ""

```

Universitas Indonesia

```


8  Browser NewPage,"HTMLTitle=Sistem Pelayanan ePaspor Departemen Luar Negeri
RI;Index=0", ""
9  EditBox Click, "Name=ctl00$ContentPlaceHolder1$Wizard1$txtFullName", "Coords=61,10"
10 InputKeys "Rina Violyta{TAB}120500080Y"
11 ComboBox Click, "Name=ctl00$ContentPlaceHolder1$Wizard1$ddl_passport_types", ""
12 ComboBox Click, "Name=ctl00$ContentPlaceHolder1$Wizard1$ddl_passport_types",
"Text=Diplomatik"

13 End Sub

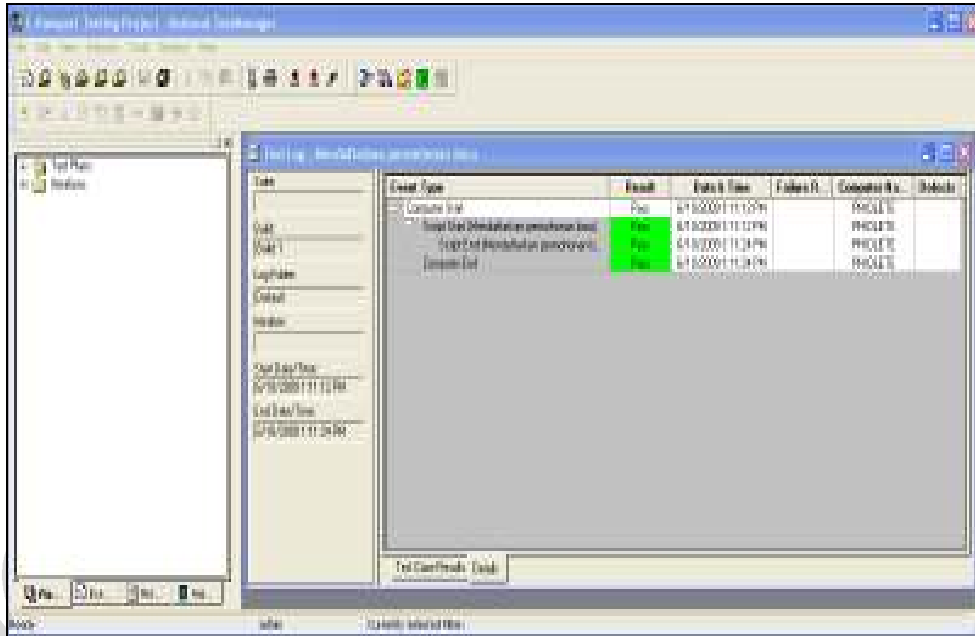
```

Aksi pertama, yakni klik pada editbox Nama Lengkap, diubah oleh Rational Robot menjadi *script* 1 baris 9. Baris pada *script* tersebut menunjukkan objek yang direkam adalah sebuah *editbox* yang bernama “ctl00\$ContentPlaceHolder1\$Wizard1\$txtFullName” yang terletak pada koordinat 61 dan 10 relatif dari ukuran layar. Aksi yang kedua, yaitu menginput “Rina Violyta”, kemudian tab, dan menginput lagi “120500080Y” ditunjukkan pada *script* 1 baris 10. Sedangkan aksi yang terakhir, yaitu memilih nilai dari *combolistbox* ditunjukkan oleh baris 11 dan 12.

Setelah *script* berhasil dibuat, maka langkah selanjutnya adalah dengan mengulangi interaksi terhadap E-Paspor yang dilakukan selama perekaman. Proses ini dinamakan dengan *playback script*. Sebelum melakukannya, pastikan *window* yang aktif hanya Rational Robot dan *browser* E-Paspor. Selama *playback* berlangsung, tidak disarankan untuk melakukan interaksi apapun dengan menggunakan *keyboard* atau *mouse*. Cara untuk melakukan *playback* tersebut adalah:

1. Pada Rational Robot, tekan tombol  yang terletak pada *toolbar*
2. Pilih *script* mana yang akan diulang, dalam hal ini “Mendaftarkan permohonan baru”, dan klik OK
3. Pilih di folder mana log hasil *playback* akan disimpan dan nama log-nya, dan klik OK

4. Rational Robot memulai proses *playback*. Setelah proses *playback* selesai, akan muncul Rational TestManager yang menampilkan log hasil *playback* seperti yang terlihat pada gambar berikut:



Gambar 4.7 Log Hasil *Playback*

4.2 Pemanfaatan *Datapool*

Setelah mengetahui bagaimana membuat *test script* secara otomatis dengan Rational Robot, langkah selanjutnya adalah dengan memperkaya uji fungsionalitas dengan menggunakan *datapool*.

4.2.1 Pengertian *Datapool*

Datapool merupakan sebuah kumpulan data-data pengujian [URR]. *Datapool* inilah yang memberikan nilai-nilai pada suatu script sehingga pada saat Robot melakukan *playback*, nilai yang dimasukkan ke dalam sistem yang diuji dapat berubah-ubah. Mengingat pada saat *playback*, Robot hanya mengulang kegiatan yang dilakukan pada saat perekaman (*record GUI*) termasuk nilai-nilai yang dimasukkan ke dalam sistem yang diuji.

Demi penjelasan yang lebih konkrit, penulis menjelaskan apa manfaat *datapool* dengan menggunakan sebuah contoh. Salah satu contoh dari penggunaan *datapool* ini misalnya pada pengujian fungsi pendaftaran sebuah permohonan. Misalnya

permohonan yang didaftarkan pada aplikasi E-Paspor adalah sebanyak 100 buah. Pendaftaran permohonan ini bisa dilakukan dengan melakukan *playback* dari *script* yang telah dibuat sebanyak 100 kali tanpa menggunakan *datapool*. Namun, keseratus permohonan yang dihasilkan akan memiliki data yang sama persis. Hal ini dapat diatasi dengan menggunakan *datapool* sebagai penyedia data yang dinamis untuk dimasukkan ke dalam sistem yang diuji pada saat proses *playback* dalam satu *script*. Sehingga dalam contoh ini dapat disimpulkan bahwa dengan adanya *datapool*, satu *script* dapat membuat 100 permohonan dengan data yang berbeda-beda.

4.2.2 Pengertian Tipe Data

Tipe data adalah sumber data untuk dimasukkan pada suatu kolom dalam *datapool*[URR]. *Datapool* terbentuk dari kolom-kolom yang merepresentasikan atribut dari *datapool* itu sendiri. Berdasarkan contoh di atas, misalkan data-data yang dibutuhkan dalam pendaftaran permohonan adalah data-data pemohon sehingga *datapool* yang dibutuhkan adalah *datapool* pemohon. Kolom-kolom yang dibutuhkan antara lain nama depan, nama belakang, nomor identitas (KTP, KTM, dan lain-lain), alamat, nomor telepon, dan sebagainya. Masing-masing kolom memiliki tipe datanya masing-masing (*data type*). Tipe data dari nama depan bisa diambil dari tipe data yang telah didefinisikan di Rational (disebut *standard data type*), yakni First-Name, atau tipe data yang dibuat sendiri. Pembuatan tipe data ini akan dijelaskan kemudian. Masing-masing tipe data memiliki data-datanya masing-masing. Data dari tipe data First-Name antara lain Koganti, Clarence, Pam, Izzy, dan lain-lain. Dengan adanya tipe data ini, dapat dihasilkan sejumlah baris data dalam suatu *datapool*.

4.2.3 Proses Pembuatan *Datapool*

Pembuatan dan pengaturan *datapool* dilakukan dengan menggunakan Rational TestManager. Sedangkan penggunaannya adalah di *script* yang ada di Rational Robot. Proses pembuatan *datapool* ini membutuhkan beberapa langkah sebagai berikut[URR]:

4.2.3.1 Perencanaan Pembuatan *Datapool*

Langkah pertama dalam pembuatan *datapool* adalah merencanakan kolom apa saja yang dibutuhkan. Kolom-kolom inilah yang membentuk *datapool*. Selain itu, hal yang perlu direncanakan adalah pemilihan tipe data yang sesuai dan perlukah membuat tipe data (*data type*) baru di luar tipe data yang sudah didefinisikan oleh Rational.

Proses identifikasi kolom-kolom yang diperlukan dalam *datapool* bisa dilakukan dengan melihat GUI pada sistem yang sedang diuji. Dalam studi kasus aplikasi E-passport, dicontohkan pada GUI pendaftaran pemohon paspor yang digunakan oleh pengguna Staf Loker Penerimaan. Pengajuan permohonan bisa berupa pembuatan paspor baru dengan kombinasi permohonan exit permit dan rekomendasi visa atau permohonan berdasarkan paspor yang sudah dimiliki. Pada opsi pertama, yakni permohonan pembuatan paspor baru dengan kombinasi permohonan exit permit dan rekomendasi visa, dimaksudkan bahwa apabila pemohon mengajukan permohonan paspor baru maka ia dapat mengajukan permohonan exit permit dan rekomendasi visa, exit permit saja, rekomendasi visa saja, atau tidak keduanya (hanya pengajuan paspor baru saja). Berikut ini gambar GUI aplikasi E-passport untuk proses pendaftaran pemohon:

Gambar 4.8 GUI Aplikasi E-passport: Pendaftaran Pemohon untuk Paspor Baru

Pada opsi tersebut, terlihat di GUI bahwa nilai yang harus dimasukkan adalah nama pemohon dan nomor identitasnya. Dua hal ini bisa dijadikan kolom-kolom di datapool. Berikut salah satu contoh kolom-kolom yang dapat memenuhi nilai-nilai tersebut. Nama pemohon dapat terdiri dari dua nama, nama depan nama belakang. Dengan demikian, datapool dibuat memiliki dua kolom, misalkan nama kolomnya **nama_depan** dan **nama_belakang**. Data-data untuk kedua kolom ini dapat dihasilkan dengan memilih tipe data Names-First dan Names-Last yang sudah didefinisikan, yang dinamakan *standard data type*. Kolom ketiga digunakan untuk menghasilkan nilai-nilai berupa nomor identitas, misalkan kolom ini dinamakan **identitas**. Karena nomor identitas dapat terdiri dari angka dan huruf, maka dapat memanfaatkan tipe data standar yaitu Random Alphanumeric String. Selain nama pemohon dan nomor identitas yang dibutuhkan agar nilainya berubah-ubah, dari GUI terlihat bahwa jenis paspor juga bisa dipilih antara jenis paspor dinas dan jenis paspor diplomatik. Berdasarkan hal ini, tidaklah salah apabila dibuat kolom keempat untuk menampung kedua nilai tersebut (dinas dan diplomatik), misalkan namanya **jenis_paspor**. Oleh karena nilai-nilai yang akan dihasilkan adalah dinas dan diplomatik yang tidak didefinisikan di *standard data type*, maka perlu juga dibuat tipe data baru sebagai penampung kedua nilai

Universitas Indonesia

tersebut. Nama untuk tipe data baru ini misalkan **Jenis Paspor**. Pada GUI, nilai-nilai yang menjadi pertimbangan dibuatnya kolom-kolom berikutnya adalah kombinasi permohonan exit permit dan rekomendasi visa seperti yang telah dijelaskan pada paragraf sebelumnya. Keempat kombinasi tersebut dapat dihasilkan dengan membuat dua kolom tambahan, yakni misalkan **permintaan_ep** dan **permintaan_rv** dengan tipe data Boolean yang nilainya true dan false. Tipe data ini tidak didefinisikan di *standard data type* sehingga harus didefinisikan sendiri, misalkan dengan nama **Boolean**. Bagaimana cara membuat datapool dan tipe data ini akan dibahas pada langkah berikutnya.

4.2.3.2 Pembuatan Kode *Script*

Setelah kolom-kolom dan tipe data dari datapool yang akan dibuat dianalisis, langkah berikutnya adalah membuat kode *script* pada Rational Robot.

Langkah awal pembuatan *script* ini adalah melakukan perekaman (*recording*) dengan Rational Robot. Objek-objek yang direkam diutamakan pada objek yang inputnya membutuhkan data dari datapool, misalnya editbox, combobox, combolistbox, dan lain-lain. Disamakan dengan contoh sebelumnya yakni pada studi kasus aplikasi E-passport, berikut ini contoh gambar GUI beserta input yang dimasukkan ke dalam objek-objek dalam GUI dan *script* hasil rekaman pada GUI aplikasi E-passport untuk pendaftaran permohonan paspor baru:

Gambar 4.9 GUI E-passport dan Inputnya

Script 2

```

1 Sub Main
2   Dim Result As Integer
3
4   'Initially Recorded: 3/17/2009 8:29:57 AM
5   'Script Name: mentahan_daftar
6
7   Window SetContext, "Caption=Sistem Pelayanan ePaspor Departemen Luar Negeri RI -
Microsoft Internet Explorer", ""
8   Browser NewPage,"HTMLTitle=Sistem Pelayanan ePaspor Departemen Luar Negeri
RI;Index=0", ""
9
10      EditBox Click, "Name=ctl00$ContentPlaceHolder1$Wizard1$txtFullName",
"Coords=48,15"
11   InputKeys "Rina Violyta"
12
13   EditBox Click, "Name=ctl00$ContentPlaceHolder1$Wizard1$txtIdentityNo",
"Coords=47,11"
14   InputKeys "KTM120500080Y"
15

```

```

16  ComboBox Click, "Name=ctl00$ContentPlaceHolder1$Wizard1$ddl_passport_types", ""
17  ComboBox Click, "Name=ctl00$ContentPlaceHolder1$Wizard1$ddl_passport_types",
"Text=Dinas"
18
19  CheckBox Click, "Name=ctl00$ContentPlaceHolder1$Wizard1$cbx_exitpermit"
20  CheckBox Click, "Name=ctl00$ContentPlaceHolder1$Wizard1$cbx_rekvisa"
21
22  PushButton Click,
"Name=ctl00$ContentPlaceHolder1$Wizard1$StartNavigationTemplateContainerID$StartNe
xtButton"
23  Browser NewPage,"HTMLTitle=Sistem Pelayanan ePaspor Departemen Luar Negeri
RI;Index=0", ""
24  PushButton Click,
"Name=ctl00$ContentPlaceHolder1$Wizard1$FinishNavigationTemplateContainerID$Fi
nishButton"

25 End Sub

```

Perhatikan gambar 1 dan *script* 2. Penjelasan berikutnya akan menggunakan kedua hal tersebut.

Pada GUI terlihat informasi kepemilikan e-paspor dipilih untuk pilihan “belum memiliki e-paspor” sehingga isian yang memerlukan input adalah “Nama Lengkap”, “No. Identitas”, jenis paspor (dinas atau diplomatik), dan *checkbox* permohonan Exit-Permit serta Rekomendasi Visa. Isian untuk editbox “Nama Lengkap” adalah “Rina Violyta” yang diubah dalam *script* menjadi baris 10-11 pada *script* 1. Demikian juga dengan isian pada editbox “No. Identitas” sama dengan *script* baris 13-14, kemudian pilihan pada combobox jenis paspor (dinas atau diplomatik) sama dengan *script* baris 16-17, dan aksi klik pada checkbox Exit-Permit serta Rekomendasi Visa sama dengan *script* baris 19-20. Nilai “Rina Violyta” (baris 11), “KTM120500080Y” (baris 14), “Dinas” (baris 17) dapat diganti dengan variabel yang menampung nilai-nilai baru yang diambil dari datapool. Sedangkan aksi yang dilakukan dengan klik *checkbox* permohonan Exit-Permit dan Rekomendasi Visa tidak menampung nilai dari *checkbox* tersebut

(apakah *checkbox* tersebut *checked* atau *unchecked*) sehingga tidak bisa mengubah objek *checkbox* tersebut menjadi *checked* atau sebaliknya secara otomatis dengan menggunakan datapool. Dengan demikian, perancangan datapool yang dilakukan pada langkah 1 dapat direvisi dengan meniadakan kolom **permintaan_rv** dan **permintaan_ep**, demikian juga dengan tipe data **Boolean** karena tidak diperlukan.

Berikut ini akan dijelaskan bagaimana cara mengambil data-data dalam datapool dan menggantikan nilai-nilai yang telah disebutkan sebelumnya dengan variabel. Perhatikan *script* 3 berikut:

Script 3

```

1 '$include "squtil.sbh"
2
3 Sub Main
4 Dim Result As Integer
5
6 'Initially Recorded: 3/16/2009 8:25:56 AM
7 'Script Name: daftar
8
9 Dim dp as Long 'Variabel untuk akses ke datapool
10 Dim x as Integer 'Variabel untuk loop
11
12 'Variabel yang digunakan untuk memasukkan data dari datapool
13 Dim nmdepan as String
14 Dim nmbelakang as String
15 Dim identitas as String
16 Dim jenisPaspor as String
17
18 'Membuka datapool Applicants
19 dp = SQADatapoolOpen("Applicants")
20
21 'Mengeksekusi transaksi sebanyak 4 kali (mendaftarkan 4 pemohon)
22 For x = 0 to 3

```

```

23
24 'Melakukan fetching ke baris pertama di datapool
25 Call SQADatapoolFetch(dp)
26
27 Window SetContext, "Caption=Sistem Pelayanan ePaspor Departemen Luar Negeri RI
- Microsoft Internet Explorer", ""
28 Browser NewPage,"HTMLTitle=Sistem Pelayanan ePaspor Departemen Luar Negeri
RI;Index=0", ""
29
30 'Bagian berikut menggunakan data dari datapool Applicants
31
32 'Nama lengkap (nama depan dan nama belakang)
33   EditText   Click,   "Name=ctl00$ContentPlaceHolder1$Wizard1$txtFullName",
"Coords=91,6"
34 Call SQADatapoolValue(dp,1,nmdepan)
35 Call SQADatapoolValue(dp,2,nmbelakang)
36 InputKeys nmdepan+" "+nmbelakang
37
38 'No. Identitas
39 EditText Click, "Name=ctl00$ContentPlaceHolder1$Wizard1$txtIdentityNo",
"Coords=86,2"
40 Call SQADatapoolValue(dp,3,identitas)
41 InputKeys identitas
42
43 'Jenis Paspor
44 ComboBox Click, "Name=ctl00$ContentPlaceHolder1$Wizard1$ddl_passport_types",
""
45 Call SQADatapoolValue(dp,4,jenisPaspor)
46           ComboBox                               Click,
"Name=ctl00$ContentPlaceHolder1$Wizard1$ddl_passport_types", "Text=" + jenisPaspor
47
48 PushButton Click,
"Name=ctl00$ContentPlaceHolder1$Wizard1$StartNavigationTemplateContainerID$Star
tNextButton"
49 Browser NewPage,"HTMLTitle=Sistem Pelayanan ePaspor Departemen Luar Negeri
RI;Index=0", ""

```

```

50 PushButton Click,
"Name=ctl00$ContentPlaceHolder1$Wizard1$FinishNavigationTemplateContainerID$Fi
nishButton"
51
52 Next x      'Melanjutkan ke loop berikutnya
53
54 Call SQADatapoolClose(dp)
55
56 End Sub

```

Sebelum menggunakan perintah-perintah *datapool*, dalam *script* harus didefinisikan terlebih dahulu *header*-nya, yakni **squtil.sbh** seperti ditunjukkan pada *script* 3 baris ke-1. Setelah itu, perintah-perintah *datapool* seperti SQADatapoolOpen, SQADatapoolFetch, SQADatapoolValue, dan SQADatapoolClose dapat digunakan. Perintah SQADatapoolOpen berfungsi untuk membuka sebuah *datapool* yang namanya dimasukkan sebagai parameter seperti dicontohkan pada baris ke-19. Perintah ini mengembalikan nilai dalam tipe data Long sehingga harus ditampung dalam variabel yang juga bertipe Long, dalam contoh di atas adalah variabel dp. Perintah SQADatapoolFetch digunakan untuk mendapatkan seluruh data pada baris pertama dari *datapool*. Perintah ini dieksekusi dengan menjalankan fungsi Call seperti yang terlihat pada baris 25. SQADatapoolClose merupakan perintah *datapool* untuk menghentikan akses ke *datapool* yang telah dibuka. Sedangkan SQADatapoolValue berfungsi untuk mendapatkan nilai pada suatu kolom dari baris data yang telah didapatkan dengan perintah SQADatapoolFetch. SQADatapoolValue membutuhkan tiga parameter. Penggunaan SQADatapoolValue diperlihatkan pada baris 34, 35, 40, dan 45. Parameter pertama adalah akses ke *datapool*, dalam contoh ini adalah variabel dp, kemudian parameter kedua adalah nomor kolom dimana data yang akan diambil berada, dan parameter ketiga adalah variabel untuk menampung data yang sudah diambil dari kolom yang sudah ditentukan. Selain nomor kolom, parameter kedua dapat diisi dengan nama kolom yang ingin diambil datanya, seperti dicontohkan sebagai berikut:

Script 4

```
'No. Identitas
EditBox Click, "Name=ctl00$ContentPlaceHolder1$Wizard1$txtIdentityNo",
"Coords=86,2"
Call SQADatapoolValue(dp,"identitas",identitas)
InputKeys identitas
```

Script 4 merupakan modifikasi dari *script 3* baris 38-41 dengan mengganti parameter kedua dari perintah SQADatapoolValue. Dapat dilihat pada *script 3*, kolom yang diambil datanya adalah kolom 3 yang pada tahap perancangan telah ditentukan namanya, yakni **identitas** sehingga parameter kedua dari perintah tersebut dapat diganti dengan "identitas". Apabila yang digunakan untuk mengisi parameter kedua dari perintah SQADatapoolValue adalah nama kolom, maka harus dipastikan isi parameter tersebut sama dengan nama kolom pada datapool yang berkaitan.

Setelah mengerti fungsi dari beberapa perintah datapool, penjelasan berikutnya adalah mengenai penggantian nilai yang dimasukkan ketika proses rekaman dengan variabel yang menampung data dari datapool. Bandingkan kedua *script* berikut yang dicuplik dari *script 2* dan *script 3*.

Cuplikan *script 2*

```
10 EditBox Click, "Name=ctl00$ContentPlaceHolder1$Wizard1$txtFullName",
"Coords=48,15"
11 InputKeys "Rina Violyta"
```

Cuplikan *script 3*

```
32 'Nama lengkap (nama depan dan nama belakang)
```



```

33  EditText  Click,  "Name=ctl00$ContentPlaceHolder1$Wizard1$txtFullName",
"Coords=91,6"
34  Call SQADatapoolValue(dp,1,nmdepan)
35  Call SQADatapoolValue(dp,2,nmbelakang)
36  InputKeys nmdepan+" "+nmbelakang

```

Nilai “Rina Violyta” pada *script* 2 diganti dengan variabel **nmdepan** + “ ” + **nmbelakang** seperti terlihat pada *script* 3. Variabel **nmdepan** menampung data dari kolom 1 datapool Applicants pada baris tertentu, yang artinya nama pertama dari seorang pemohon (misalnya Rina dari nama lengkap Rina Violyta). Sedangkan variabel **nmbelakang** menampung data dari kolom kedua datapool Applicants pada baris yang sama, yang berarti nama kedua dari seorang pemohon (misalnya Violyta dari nama lengkap Rina Violyta). Kedua variabel ini digunakan karena nama lengkap terdiri dari nama depan dan nama belakang yang didapatkan dari dua kolom yang berbeda di datapool Applicants. Proses penampungan ini dilakukan dengan memanggil perintah SQADatapoolValue seperti pada *script* 3 baris 34 dan 35. Sebelumnya, kedua variabel ini dan variabel lain yang dibutuhkan untuk menampung data dari datapool harus didefinisikan sebagai variabel sebelum digunakan, caranya adalah dengan menggunakan perintah Dim seperti terlihat pada *script* 3 baris 13-16.

4.2.3.3 Pembuatan *Datapool* dan Nilai-nilai di Dalamnya

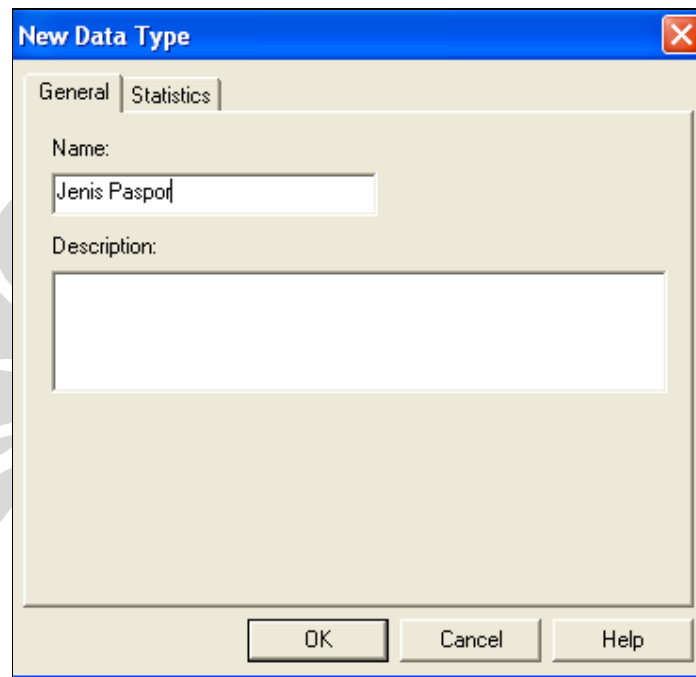
Perancangan *datapool* mengenai kolom-kolom yang diperlukan serta tipe data apa saja yang sesuai telah dibahas dan telah ditentukan untuk studi kasus aplikasi E-Paspor. Dengan demikian, tahap selanjutnya adalah mengimplementasikan rancangan yang telah dibuat tersebut. *Datapool* dibuat dan diatur dengan menggunakan Rational TestManager. Sebelumnya, akan dijelaskan bagaimana cara membuat tipe data karena dalam studi kasus ini dibutuhkan tipe data di luar tipe data standard untuk kemudian digunakan dalam pembuatan *datapool*.

1. Pembuatan Tipe Data

Berdasarkan perancangan yang telah dilakukan terhadap studi kasus, yakni aplikasi E-Passport, tipe data yang perlu dibuat secara manual adalah **Jenis**

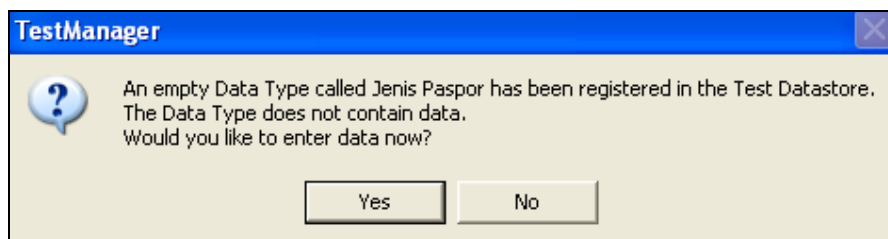
Paspor. Berikut ini adalah langkah-langkah pembuatannya dengan menggunakan Rational TestManager:

1. Pilih menu *Tools – Manage – Data Types*.
2. Pilih *New* kemudian masukkan nama tipe data (dalam contoh ini nama tipe data yang dimasukkan adalah Jenis Paspor), lalu pilih *Ok*



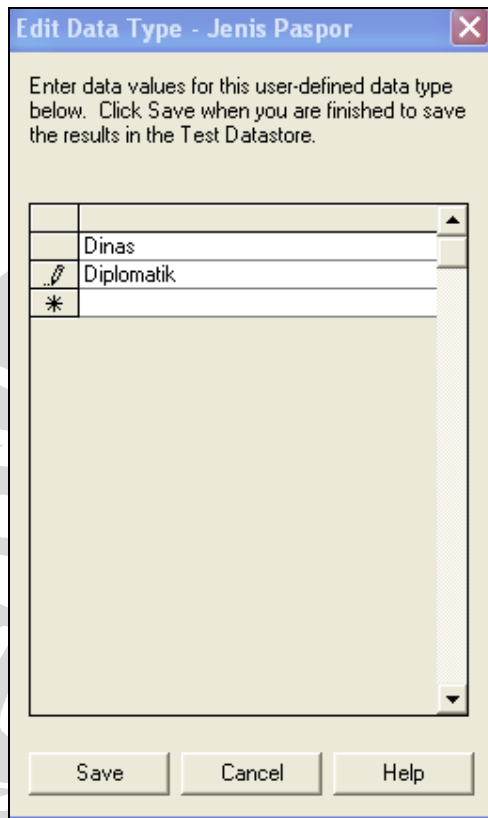
Gambar 4.10 Membuat *Data Type* Baru

3. Muncul window baru yakni konfirmasi apakah data-data untuk tipe data dengan nama Jenis Paspor ini akan dimasukkan. Pilih *Yes*.



Gambar 4.11 *Window* Konfirmasi Pengisian Data untuk *Data Type* Baru

4. Masukkan data-data yang diperlukan. Dalam studi kasus ini hanya terdapat dua jenis paspor, yakni dinas dan diplomatik, sehingga data yang dimasukkan adalah Dinas yang diletakkan di baris pertama dan Diplomatik yang dimasukkan di baris kedua.



Gambar 4.12 Pengisian Data untuk *Data Type* Baru

5. Pilih *Save*

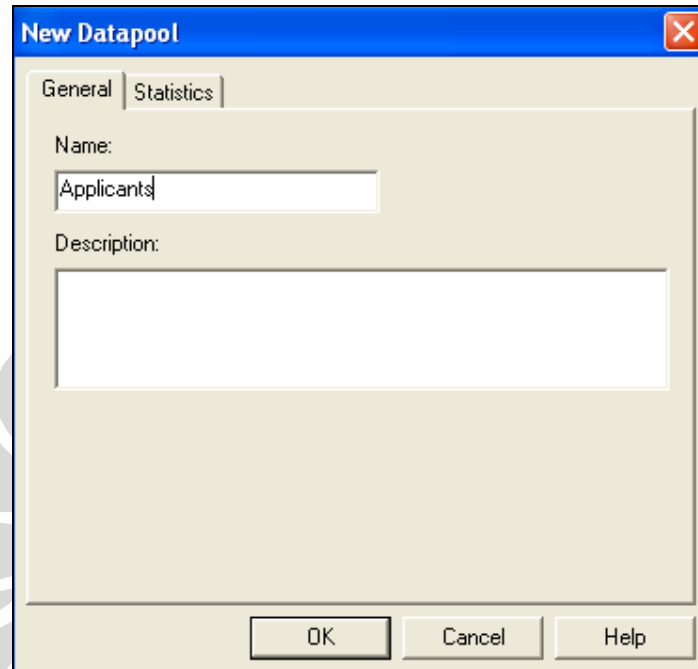
Dengan demikian, tipe data baru telah selesai dibuat. Tipe data ini kemudian dapat di-*rename*, dihapus, disalin, dan diedit nilainya. Selanjutnya akan dibuat *datapool* yang menggunakan nilai-nilai dalam tipe data-tipe data yang telah dibuat.

2. Pembuatan *Datapool*

Dengan menggunakan Rational TestManager, berikut ini tahap-tahap dalam pembuatan *datapool*:

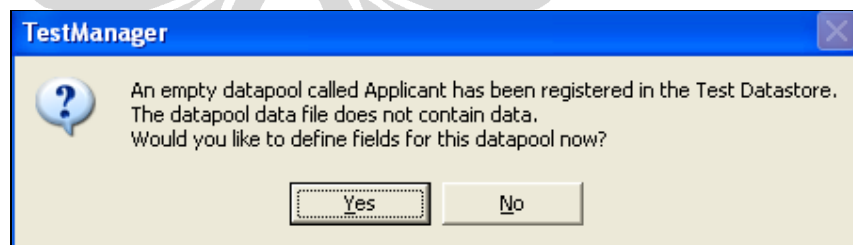
1. Pilih menu *Tools – Manage – Datapools*

- Pilih *New* kemudian masukkan nama datapool. Karena telah ditetapkan dalam tahap perancangan, nama untuk datapool ini adalah *Applicants*. Kemudian pilih *Ok*.



Gambar 4.13 Pembuatan *Datapool* Baru

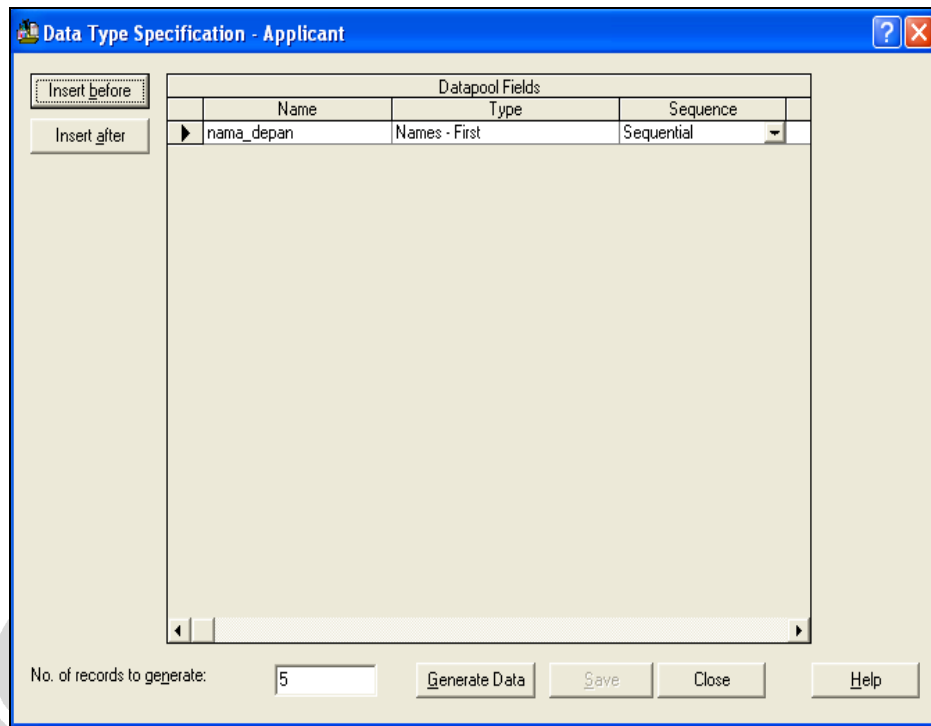
- Muncul window baru yang meminta konfirmasi apakah kolom-kolom untuk datapool tersebut akan dibuat. Pilih *Yes*.



Gambar 4.14 *Window* Konfirmasi Pengisian Nama Kolom *Datapool*

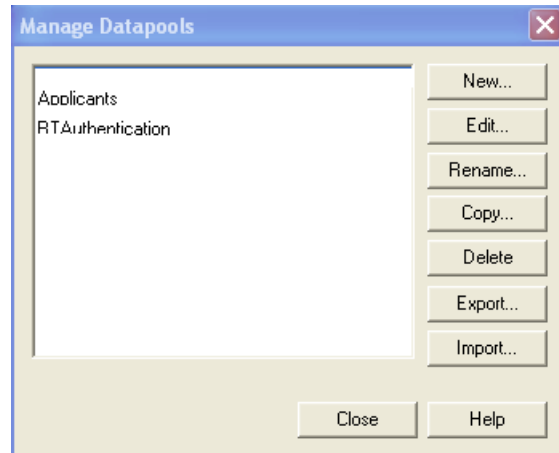
- Klik *Insert before* atau *Insert after* untuk mendefinisikan kolom-kolom. Tiap baris merepresentasikan satu kolom dalam *datapool*.
- Masukkan nama kolom yang pertama (yakni nama_depan), kemudian tipe datanya (yaitu *Names - First*), dan *property* yang lain.

Universitas Indonesia



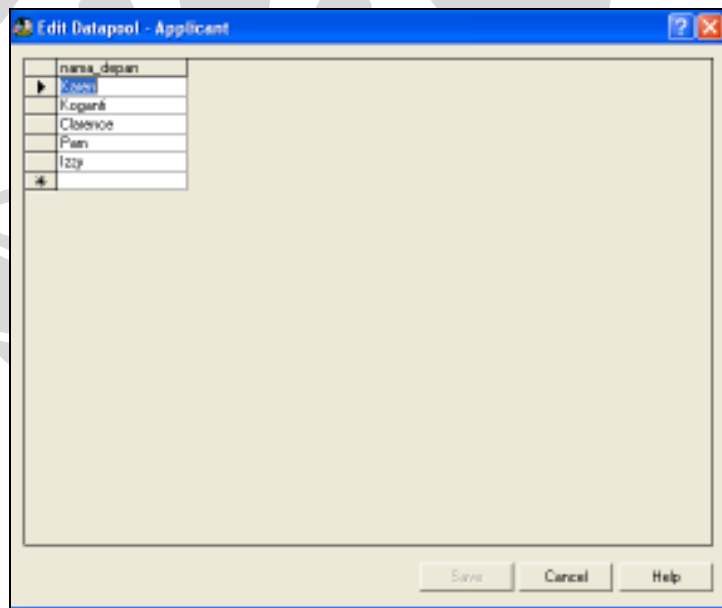
Gambar 4.15 Pengisian Nama Kolom Beserta Propertinya

6. Ulangi tahap 4 hingga tahap 5 sampai semua kolom didefinisikan.
7. Masukkan jumlah baris data yang ingin dibuat dalam *editbox* “No. records to generate”
8. Klik *Generate Data* setelah semua didefinisikan.
9. Pilih *Yes* untuk melihat ringkasan data yang telah dihasilkan
10. Untuk melihat data-data yang ada dalam *datapool* yang telah dibuat, tutup *window* yang digunakan untuk mendefinisikan kolom-kolom *datapool* hingga muncul *window* seperti berikut:



Gambar 4.16 Window Pengaturan *Datapool*

11. Pilih nama *datapool* kemudian tekan Edit
12. Pilih *Edit Datapool Data* sehingga muncul *window* yang memunculkan nilai-nilai *datapool*, yaitu nama depan, yang telah di-generate



Gambar 4.17 Nilai-nilai dalam *Datapool* Setelah Di-generate

Dengan menyelesaikan pembuatan *datapool*, maka dapat diujicobakan pemanfaatan *datapool* ini dengan cara melakukan *playback* pada *script* yang telah dimodifikasi dengan menggunakan perintah-perintah *datapool*.

Universitas Indonesia

4.3 Penggunaan *Verification Point*

Verification point merupakan titik verifikasi untuk memastikan status dari suatu objek. Menggunakan Rational Robot dapat merekam penelusuran suatu aplikasi. *Script* yang dihasilkan sangat berguna untuk memastikan bahwa aplikasi dapat berjalan. Akan tetapi, hal tersebut belum dikatakan sebagai sebuah pengujian karena belum ada proses verifikasi apapun. Misalnya pada saat login dengan menggunakan *username* Administrator, titik *verification point* dapat ditempatkan setelah proses login dieksekusi untuk memastikan bahwa Administrator telah berhasil login.

4.3.1 Jenis-jenis *Verification Point*

Terdapat beberapa macam *verification point* dalam Rational Robot yang bisa dimanfaatkan. Semuanya dapat dilihat dengan memilih menu *Insert* → *Verification Point* pada Rational Robot. Masing-masing *verification point* dijelaskan dalam tabel berikut:


Tabel 4.1 Jenis *Verification Point*

Nama	Deskripsi
<i>Alphanumeric</i>	Memverifikasi data alfanumerik di dalam suatu <i>edit box</i> , tombol, label, <i>text area</i> , dan sebagainya
<i>Clipboard</i>	Memverifikasi konten dari <i>Windows clipboard</i>
<i>Menu</i>	Memverifikasi nilai-nilai dalam menu dan statusnya (<i>enabled</i> atau <i>disabled</i>)
<i>Object data</i>	Memverifikasi konten data dari suatu objek
<i>Object properties</i>	Memverifikasi <i>property</i> dari suatu objek (misalnya <i>font</i> , <i>color</i> , <i>position</i>)
<i>Region image</i>	Secara grafis membandingkan area pada layar
<i>Window existence</i>	Menguji apakah suatu <i>window</i> ada atau tidak pada layar
<i>Window image</i>	Secara grafis membandingkan keseluruhan <i>window</i> seperti <i>window box</i>

Kumpulan jenis *verification point* ini menyediakan kemampuan yang lebih untuk memverifikasi status dan data dari suatu aplikasi yang diuji. Dalam setiap kasus, dasarnya adalah dengan mendapatkan status terakhir dari suatu aplikasi yang disebut dengan *baseline*. *Baseline* ini dibandingkan dengan status aktual yang didapatkan pada saat *playback*. Apabila keduanya sama, *verification point* menghasilkan *pass*. Jika sebaliknya, maka *verification point* menghasilkan *fail*.

Sebagai contoh untuk menunjukkan bagaimana *verification point* dalam Rational Robot bekerja, berikut ini adalah langkah-langkah menambahkan *verification point* untuk memastikan bahwa yang memasuki halaman pendaftaran permohonan baru adalah Staf Loker Penerimaan dengan *username* stafloketpenerimaan. *Verification point* yang cocok untuk kasus ini adalah *Object Data*. Cara menambahkan *verification point* ke dalam *script* dijelaskan sebagai berikut:

1. Buka *script* "Mendaftarkan permohonan baru".
2. Letakkan kursor pada baris di bawah *script*:

```
"Browser NewPage,"HTMLTitle=Sistem Pelayanan ePaspor Departemen Luar Negeri RI;Index=0", ""
```
3. Pilih menu Insert → Verification Point → Object Data.
4. Muncul *window* untuk mengatur *wait state* untuk memberikan tenggang waktu dimulainya *verification point* dan *expected result*. Setelah pengaturan selesai, klik OK.
5. Muncul *window Select Object*, klik pada  dan jangan dilepas hingga menemukan objek yang diverifikasi, dalam hal ini label yang bertuliskan Staf Loker Penerimaan, lalu lepaskan.
6. Klik OK pada *window Select Object* yang muncul.
7. Pilih "Contents" pada *window Object Data Tests* yang muncul dan klik OK.
8. Kemudian muncul *window Object Data Verification Point*. Pilih *verification method* yang sesuai dan pilihan yang lain (dalam hal ini pilih yang sudah *default* saja). Klik OK.
9. Pada *script* ditambahkan *verification point* seperti berikut:

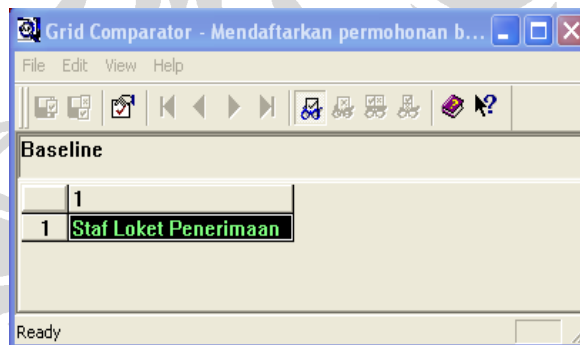
Script 5

```

1 Window SetTestContext, "Caption=Sistem Pelayanan ePaspor Departemen Luar Negeri
RI – Microsoft Internet Explorer", ""
2 Browser NewPage,"HTMLTitle=Sistem Pelayanan ePaspor Departemen Luar Negeri
RI;Index=0", ""
3 Result = HTMLVLP (CompareData, "HTMLId=ctl00_lblUserName", "VP=Object
Data;Wait=2,10")
4 Window ResetTestContext, "", ""

```

Script di atas menunjuk sebuah *baseline*, yaitu dengan VP = Object Data. Pada *window* Rational Robot, yaitu pada panel sebelah kiri, terlihat daftar *verification point* yang telah dibuat. *Verification point* Object Data terdapat di sana. Dengan melakukan *double klik* pada Object Data, akan terlihat *baseline* yang telah didapatkan seperti yang terlihat pada gambar berikut:



Gambar 4.18 *Baseline* pada *Verification Point* Object Data

Verification point di atas berfungsi membandingkan *baseline* yang terlihat pada gambar 4.18 dan nilai aktual yang ditunjukkan pada *script* 5 baris 3, yaitu pada parameter kedua “HTMLId = ctl00_lblUserName” yang nilainya akan didapatkan pada proses *playback*.

4.3.2 Kegagalan *Verification Point* untuk Data Dinamis dan Solusinya

Dalam implementasinya terhadap aplikasi E-Paspor, *verification point* tidak dapat digunakan untuk memastikan misalnya, apakah setelah pendaftaran permohonan,

data seorang pemohon sudah masuk ke dalam sistem atau belum. Hal ini dikarenakan data pemohon yang dimasukkan tersebut diambil dari *datapool* secara dinamis. Sedangkan *baseline* yang dijadikan pembanding dengan status aktual merupakan data yang statis. Hal ini dapat diatasi dengan membuat *verification point* tambahan secara manual dengan menggunakan perintah **if then...else**, **sqaGetPropertyAsString**, dan **sqaLogMessage**.

Sebuah contoh dapat menggambarkan bagaimana penggunaan ketiga perintah-perintah tersebut. Berikut ini merupakan cuplikan *script dataEntry_data* yang merupakan *verification point* manual yang ditujukan untuk memverifikasi apakah nama pemohon yang sudah didaftarkan oleh staf loket penerimaan dan sudah di-*upload* fotonya telah masuk ke daftar pemohon yang akan diisi datanya oleh staf *data entry*.

Cuplikan *script dataEntry_data*

```

1 Window SetContext, "Caption=Sistem Pelayanan ePaspor Departemen Luar Negeri RI -
Microsoft Internet Explorer", ""
2 Browser NewPage, "HTMLTitle=Sistem Pelayanan ePaspor Departemen Luar Negeri
RI;Index=0", ""
3 ResultVP =
SQAGetPropertyAsString("Type=Editbox;Name=ctl00$ContentPlaceHolder1$TextBox1", "Value"
, namaaktual)
4 if namaaktual = nmdepan+ " "+nmbelakang then
5   SQALogMessage sqaPass, "Baseline value: '"+nmdepan+ " "+nmbelakang +" = Actual
value: " + namaaktual + "'", "baseline dari datapool sama dengan aktualnya"
6 else
7   SQALogMessage sqaFail, "Baseline value: '"+nmdepan+ " "+nmbelakang +" != Actual
value: " + namaaktual + "'", "baseline dari datapool tidak sama dengan aktualnya"
8 end if

```

Perintah `SQAGetPropertyAsString` digunakan untuk mendapatkan data dari suatu objek, dalam cuplikan *script* di atas adalah data dari objek *edit box* yang bernama “`ctl00$ContentPlaceHolder1$TextBox1`”. Data ini kemudian disimpan dalam variabel **namaaktual**. Pada baris 4, nilai yang disimpan dalam variabel tersebut dibandingkan dengan “*baseline*” yang berasal dari *datapool*, yang dalam hal ini disimpan dalam variabel **nmdepan** dan **nmbelakang**. Apabila keduanya cocok, maka perintah `SQALogMessage` menuliskan pesan *Pass* (yang ditunjukkan dengan **sqPass**) ke dalam *log* hasil *playback*. Jika terjadi sebaliknya, maka perintah `SQALogMessage` menuliskan pesan *Fail* (yang ditunjukkan dengan **sqFail**) ke dalam *log* hasil *playback*. Berikut adalah gambar *log* hasil *playback script* yang berisi *verification point* manual dan *verification point* yang *built-in*.

Event Type	Result	Date & Time	Failure Reason
Computer Start	Warning	5/26/2009 3:11:13 AM	
Script Start (dataEntry-data)	Pass	5/26/2009 3:11:13 AM	
Call Script		5/26/2009 3:11:13 AM	
Script Start (dataEntry-login)	Pass	5/26/2009 3:11:13 AM	
Verification Point (Object Properties) - Object Properties)	Pass	5/26/2009 3:11:29 AM	
Playback Warning	Warning	5/26/2009 3:11:59 AM	
Log Message (Baseline value: 'Marta Bussey' = Actual value: 'Marta Bussey')	Pass	5/26/2009 3:11:59 AM	
Verification Point (Object Properties) - Object Properties)	Pass	5/26/2009 3:13:57 AM	
Playback Warning	Warning	5/26/2009 3:14:27 AM	
Log Message (Baseline value: 'Bart Lound' = Actual value: 'Bart Lound')	Pass	5/26/2009 3:14:27 AM	
Verification Point (Object Properties) - Object Properties)	Pass	5/26/2009 3:16:28 AM	
Playback Warning	Warning	5/26/2009 3:16:58 AM	
Log Message (Baseline value: 'Rhods Bekawitz' = Actual value: 'Rhods Bekawitz')	Pass	5/26/2009 3:16:58 AM	
Verification Point (Object Properties) - Object Properties)	Pass	5/26/2009 3:18:55 AM	
Playback Warning	Warning	5/26/2009 3:19:25 AM	
Log Message (Baseline value: 'Dorinda Payne' = Actual value: 'Dorinda Payne')	Pass	5/26/2009 3:19:25 AM	
Verification Point (Object Properties) - Object Properties)	Pass	5/26/2009 3:21:21 AM	
Playback Warning	Warning	5/26/2009 3:21:51 AM	
Log Message (Baseline value: 'Scott Raynor' = Actual value: 'Scott Raynor')	Pass	5/26/2009 3:21:51 AM	
Script End (dataEntry-data)	Pass	5/26/2009 3:23:40 AM	
Computer End	Warning	5/26/2009 3:23:40 AM	

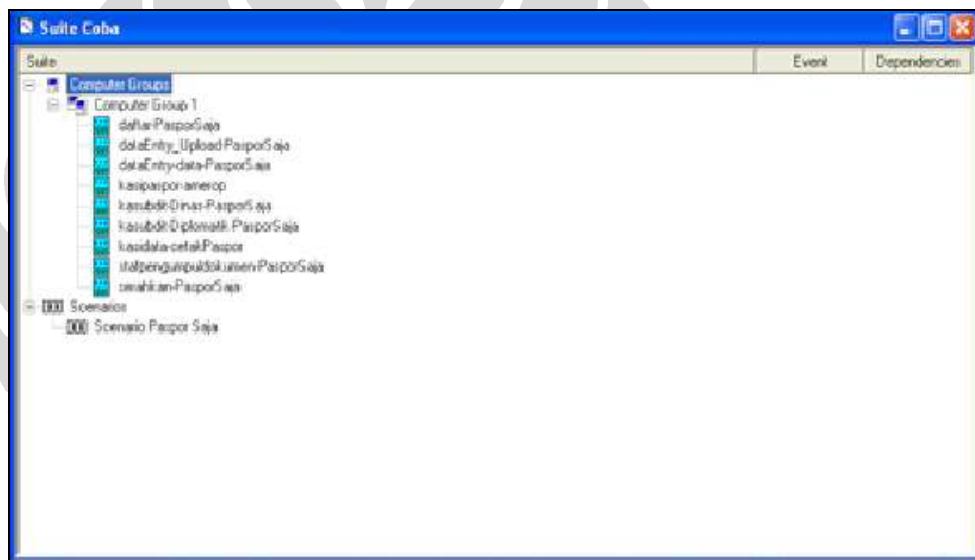
Gambar 4.19 Log yang Berisi *Verification Point* Manual dan *Built-in*

Pada gambar 4.19 terlihat *verification point* manual hasil dari penggunaan perintah **if...else**, `SQAGetPropertyAsString`, dan `SQALogMessage`. Salah satunya adalah baris ke-7 dari atas yang bertuliskan: Log Message (Baseline

value: “Marta Bussey” = Actual value: “Marta Bussey”). Karakter-karakter di dalam tanda kurung merupakan hasil dari cuplikan *script* dataEntry_data baris ke-5.

4.4 Penggunaan *Test Suite*

Test suite menyatukan beberapa *item* dalam pengujian, seperti komputer mana saja yang melakukan pengujian, *script* apa saja yang dijalankan, dan berapa kali masing-masing *script* dijalankan. Dalam pengujian kali ini, komputer yang digunakan untuk pengujian adalah satu komputer lokal. Dengan kata lain, *test suite* menggabungkan beberapa *test script* sehingga menjadi satu alur kerja yang utuh seperti yang terlihat pada gambar berikut:



Gambar 4.20 Contoh *Suite*

Sayangnya, penggunaan *test suite* di atas tidak dapat dimanfaatkan dalam Tugas Akhir ini. Penjelasannya akan dijabarkan pada subbab berikut.

4.4.1 Kegagalan Penggunaan *Test Suite*

Sebaiknya, *test script* dibuat secara modular. Berikut ini adalah daftar *test script* yang dibuat untuk menguji aplikasi E-Paspor berdasarkan *test case* yang telah dibuat pada bab dua:

Tabel 4.2 Daftar *Script* untuk E-Paspor

No	Nama <i>script</i>	No	Nama <i>script</i>	No	Nama <i>script</i>
1	penerimaan-login	16	kasipaspor-amerop-PasporSaja	31	stafrv-login
2	daftar	17	kasipasporaspasaf-login	32	stafrv-cetak
3	daftar-EPsaja	18	kasiPaspor	33	stafpengumpuldokumen-login
4	daftar-PasporSaja	19	kasipaspor-aspasaf-PasporEP	34	stafpengumpuldokumen
5	daftar-PasporEP	20	LoginKasubdit	35	stafpengumpuldokumen-PasporEP
6	dataEntry-login	21	kasubdit-dinas	36	stafpengumpuldokumen-PasporSaja
7	dataEntry_upload	22	kasubdit-dinas-PasporEP	37	stafloketsyerahan-login
8	dataEntry-data	23	kasubdit-dinas-PasporSaja	38	serahkan
9	dataEntry-data-EPsaja	24	kasubdit-diplomatik	39	serahkan-PasporEP
10	dataEntry-data-PasporEP	25	kasubdit-diplomatik-PasporEP	40	serahkan-PasporSaja
11	dataEntry-data-PasporSaja	26	kasubdit-diplomatik-PasporSaja		
12	kasipasporamerop-login	27	kasidata-login		
13	kasipaspor-amerop	28	kasidata-cetakPaspor		
14	kasipaspor-amerop-EPsaja	29	stafep-login		
15	kasipaspor-amerop-PasporEP	30	stafep-cetak		

Masing-masing *test script* di atas sudah melakukan pengujian secara terpisah tanpa ada kaitan dengan alur kerja E-Paspor. Untuk mewujudkan pengujian secara menyeluruh untuk satu alur kerja E-Paspor dari penerimaan hingga penyerahan paspor, maka digunakan *test suite*. Dari sejumlah *test script* tersebut, dapat dibuat empat alur kerja sebagai berikut:

Tabel 4.3 Daftar *Script* Berdasarkan Jenis Alur Kerja

		Jenis Alur Kerja			
		Permohonan Paspor Baru	Permohonan Paspor Baru dan Exit Permit	Permohonan Paspor Baru, Exit Permit, dan Rek. Visa	Permohonan Exit Permit
Script yang digunakan	penerimaan-login	penerimaan-login	penerimaan-login	penerimaan-login	penerimaan-login
	daftar-PasporSaja	daftar-PasporEP	daftar	daftar	daftar-EPsaja
	dataEntry-login	dataEntry-login	dataEntry-login	dataEntry-login	dataEntry-login
	dataEntry_upload	dataEntry_upload	dataEntry_upload	dataEntry_upload	dataEntry_upload
	dataEntry-data-PasporSaja	dataEntry-data-PasporEP	dataEntry-data	dataEntry-data	dataEntry-data-EPsaja
	kasipasporamerop-login	kasipasporamerop-login	kasipasporamerop-login	kasipasporamerop-login	kasipasporamerop-login
	kasipaspor-amerop-PasporSaja	kasipaspor-amerop-PasporEP	kasipaspor-amerop	kasipaspor-amerop	kasipaspor-amerop-EPsaja
	LoginKasubdit	kasipasporaspasaf-login	kasipasporaspasaf-login	kasipasporaspasaf-login	
	kasubdit-dinas-PasporSaja	kasipaspor-aspasaf-PasporEP	kasiPaspor	kasiPaspor	
	kasubdit-diplomatik-PasporSaja	LoginKasubdit	LoginKasubdit	LoginKasubdit	
	kasidata-login	kasubdit-dinas-PasporEP	kasubdit-dinas	kasubdit-dinas	
	kasidata-cetakPaspor	kasubdit-diplomatik-PasporEP	kasubdit-diplomatik	kasubdit-diplomatik	
	stafpengumpuldokumen-login	kasidata-login	kasidata-login	kasidata-login	
	stafpengumpuldokumen-PasporSaja	kasidata-cetakPaspor	kasidata-cetakPaspor	kasidata-cetakPaspor	
	serahkan-PasporSaja	stafep-login	stafep-login	stafep-login	
		stafep-cetak	stafep-cetak	stafep-cetak	
		stafpengumpuldokumen-login	stafpv-login	stafpv-login	
		stafpengumpuldokumen-PasporEP	stafpv-cetak	stafpv-cetak	
		stafloketsenyarahan-login	stafpengumpuldokumen-login	stafpengumpuldokumen-login	
		serahkan-PasporEP	stafpengumpuldokumen-serahkan	stafpengumpuldokumen-serahkan	
		stafloketsenyarahan-login	stafloketsenyarahan-login		
		serahkan	serahkan		

Pembuatan *test suite* dilakukan di Rational TestManager dengan langkah-langkah sebagai berikut:

1. Pilih menu File → New Suite → pilih Blank Functional Testing suite → OK
2. Masukkan komputer yang akan digunakan untuk pengujian dengan melakukan klik kanan pada Computer Groups dan pilih Insert → Computer Group

3. Masukkan nama komputer (gunakan *default*: Computer Group 1) dan pilih OK
4. Klik kanan pada Computer Group 1, yaitu komputer yang akan melakukan pengujian, dan pilih Insert → Test Script
5. Pilih *script* yang sesuai dengan alur kerja → OK
6. Jalankan *suite* dengan cara pilih menu File → Run Suite
7. Pilih *suite* yang akan dijalankan
8. *Suite* berjalan dengan menunjukkan diagram *progress*-nya

Pada saat *suite* menjalankan *script* yang tidak berisi perintah SQADatapoolOpen, yaitu perintah untuk membuka *datapool*, *suite* berjalan menuju *script* selanjutnya. Akan tetapi, ketika *suite* menemui *script* yang memiliki perintah tersebut di dalamnya, maka *suite* akan segera berhenti dan menghasilkan *log* pengujian yang *fail*. Pada penelitian kali ini, tidak ditemukan alasan mengapa penggunaan perintah-perintah *datapool* menyebabkan tidak berjalannya *test suite*. Hal tersebut diharapkan dapat terjawab pada penelitian selanjutnya. Singkatnya, penggunaan *suite* tidak dapat dilakukan untuk pengujian E-Paspor ini karena sebagian besar *script* menggunakan perintah-perintah *datapool* di dalamnya. Oleh karena itu, diperlukan solusi lain agar pengujian secara menyeluruh dalam satu alur kerja yang utuh tetap dapat dilaksanakan.

4.4.2 Penggunaan *CallScript* Sebagai Pengganti *Test Suite*

Perintah *callscript* digunakan untuk menggabungkan antara satu *script* dengan *script* yang lain. Perintah ini disisipkan dalam *script* yang membutuhkan *script* yang dipanggil. Dengan cara ini, maka setiap *script* dapat digabungkan sehingga membentuk alur kerja yang utuh seperti fungsi dari *test suite*. Terdapat dua cara dalam membentuk alur kerja yang utuh dengan menggunakan *callscript*, yaitu dengan membuat satu *script* tambahan untuk memanggil semua *script* secara berurutan dan dengan menyisipkan perintah *callscript* untuk memanggil *script* urutan berikutnya di akhir setiap *script*. Kedua cara ini memiliki efek yang sama dalam hal kecepatan pengujian.

Berikut ini ditampilkan satu *script* tambahan untuk melakukan pemanggilan terhadap *script-script* yang termasuk dalam masing-masing alur kerja seperti yang terlihat pada tabel 4.3.

1. *Script kumpulan-PasporSaja* untuk mengumpulkan *script-script* dengan jenis alur kerja permohonan paspor baru.

Script kumpulan-PasporSaja

Sub Main

Dim Result As Integer

'Initially Recorded: 5/26/2009 10:51:55 AM

'Script Name: kumpulan-PasporSaja

CallScript "daftar-PasporSaja"

CallScript "dataEntry_upload"

CallScript "dataEntry-data-PasporSaja"

CallScript "kasipaspor-amerop-PasporSaja"

CallScript "kasubdit-Dinas-PasporSaja"

CallScript "kasubdit-Diplomatik-PasporSaja"

CallScript "kasidata-cetakPaspor"

CallScript "stafpengumpuldokumen-PasporSaja"

CallScript "serahkan-PasporSaja"

End Sub

Script-script yang dipanggil pada *script kumpulan-PasporSaja* di atas masing-masing memanggil *script* login untuk pengguna yang sesuai, misalnya pada pemanggilan *script daftar-PasporSaja*. Pada *script daftar-PasporSaja* juga terdapat pemanggilan *script penerimaan-login* sehingga seluruh *script* yang terdaftar dalam tabel 4.3 telah dimanfaatkan dalam memenuhi pengujian untuk satu alur kerja secara utuh.

2. *Script kumpulan-PasporEP* untuk mengumpulkan *script-script* dengan jenis alur kerja permohonan paspor baru dan exit permit.

Script kumpulan-PasporEP

```

Sub Main
  Dim Result As Integer
  'Initially Recorded: 5/27/2009 2:41:01 AM
  'Script Name: kumpulan-PasporEP

  CallScript "daftar-PasporEP"
  CallScript "dataEntry_upload"
  CallScript "dataEntry-data-PasporEP"
  CallScript "kasipaspor-amerop-PasporEP"
  CallScript "kasipaspor-aspasaf-PasporEP"
  CallScript "kasubdit-dinas-PasporEP"
  CallScript "kasubdit-Diplomatik-PasporEP"
  CallScript "kasidata-cetakPaspor"
  CallScript "stafep-cetak"
  CallScript "stafpengumpuldokumen-PasporEP"
  CallScript "serahkan-PasporEP"
End Sub

```

Script-script yang dipanggil pada *script kumpulan-PasporEP* masing-masing memanggil *script* login untuk pengguna yang sesuai, misalnya pada pemanggilan *script daftar-PasporEP*. Pada *script daftar-PasporEP* tersebut terdapat pemanggilan *script penerimaan-login* sehingga seluruh *script* yang terdaftar dalam tabel 4.3 telah dimanfaatkan dalam memenuhi pengujian untuk satu alur kerja secara utuh.

3. *Script kumpulan* untuk mengumpulkan *script-script* dengan jenis alur kerja permohonan paspor baru, exit permit, dan rekomendasi visa.

Script kumpulan

Sub Main

Dim Result As Integer

'Initially Recorded: 5/14/2009 7:56:32 AM

'Script Name: kumpulan

CallScript "daftar"

CallScript "dataEntry_upload"

CallScript "dataEntry-data"

CallScript "kasipaspor-amerop"

CallScript "kasiPaspor"

CallScript "kasubdit-dinas"

CallScript "kasubdit-Diplomatik"

CallScript "kasidata-cetakPaspor"

CallScript "stafep-cetak"

CallScript "stafrv-cetak"

CallScript "stafpengumpuldokumen"

CallScript "serahkan"

End Sub

Perlu diketahui bahwa masing-masing *script* yang dipanggil pada *script* **kumpulan** juga telah memanggil *script* login, seperti **penerimaan-login**, **dataEntry-login**, dan lain sebagainya, pada *script*-nya masing-masing.

4. Pada alur kerja yang keempat, yakni permohonan untuk exit permit saja dengan menggunakan paspor lama, tidak dibentuk satu *script* tambahan untuk mengumpulkan *script-script* yang terlibat. Pengujian tidak dapat memenuhi alur kerja ini secara utuh. Hal ini dikarenakan pada saat melakukan *playback* secara individual untuk *script* **kasipaspor-amerop-EPsaja** ditemukan kerusakan pada E-Paspor yang mengakibatkan bisnis proses permohonan exit permit terhenti hingga kasipaspor. Dengan demikian, *script* yang mengimplementasikan alur kerja tersebut hanya bisa dibuat hingga *script* **kasipaspor-amerop-EPsaja**. Kerusakan ini akan dibahas pada bab Hasil Eksperimen dan Analisis.

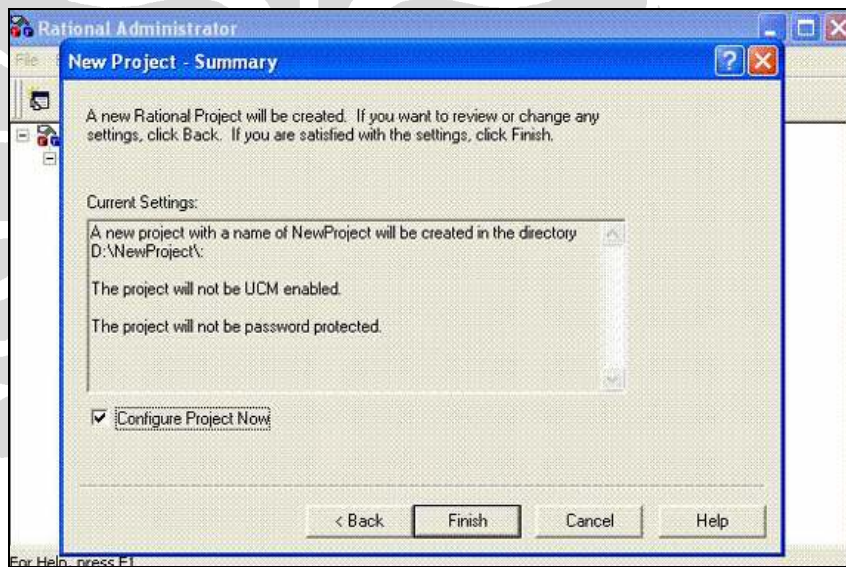
Universitas Indonesia

4.5 Pengalaman Teknis dalam Implementasi

Berikut ini akan dijelaskan mengenai pengalaman-pengalaman teknis yang juga berisi saran-saran yang dapat diberikan dalam proses implementasi pengujian.

1. Penggunaan *browser* Internet Explorer 6 untuk membuka sistem E-Paspor yang dirancang untuk *web-based*. Sistem E-Paspor akan tetap dapat dibuka pada *browser* lain, misalnya Mozilla Firefox, Internet Explorer 7, Netscape, dan Safari, dan lain-lain. Akan tetapi, objek-objek di dalamnya, seperti tombol, *text area*, *radio button*, dan lain-lain, tidak dapat dikenali oleh Rational Robot pada saat perekaman.
2. Penggunaan tombol tab untuk perpindahan dari satu objek ke objek berikutnya. Hal ini lebih baik untuk dilakukan dibandingkan dengan menggunakan klik (*mouse click*) langsung pada objek yang dituju. Robot mencatat posisi relatif suatu objek yang diklik terhadap resolusi layar apabila digunakan klik dalam proses perekaman. Dengan demikian, apabila resolusi layar berubah atau *window browser* yang digunakan untuk membuka sistem yang diuji berubah ukurannya, maka pada saat *playback*, Robot tidak akan menemukan objek yang dituju pada saat proses perekaman. Hal tersebut akan menghasilkan *log* yang gagal (*fail*) meskipun tidak ada *defect* yang dialami oleh sistem.
3. Pada saat menggunakan perintah `SQAGetPropertyAsString`, nama objek yang di-*capture* dan nama *property* yang akan diambil *value*-nya sudah diketahui secara pasti. Hal ini bisa dilakukan dengan menggunakan *Object Inspector* yang terdapat dalam menu Rational Robot. Fungsi ini dapat menangkap seluruh *property* yang dimiliki oleh suatu objek. Penggunaan `SQAGetPropertyAsString` untuk mengambil nilai dari suatu *property* tentu harus menyesuaikan dengan nama *property* yang benar sesuai dengan yang ditangkap oleh *Object Inspector*.
4. Penggunaan *verification point* dengan perintah `if...else` mempengaruhi kecepatan pengujian. Hal tersebut dikarenakan adanya proses membandingkan antara suatu nilai dengan nilai yang lain sehingga banyak memakan waktu pengujian dan menghasilkan *playback* dengan waktu yang panjang.

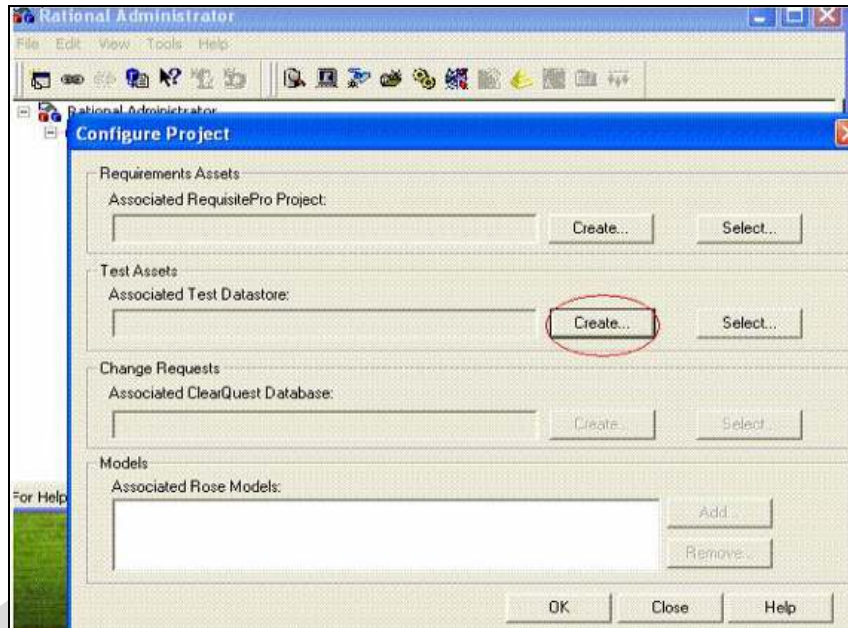
5. Apabila diperlukan penyalinan sebuah *Rational Suite Project* (.rsp) untuk kemudian dijalankan di direktori yang berbeda, atau kemudian dipindahkan ke komputer dengan spesifikasi yang lebih tinggi, prosesnya tidak cukup hanya dengan *copy* dan *paste* saja. Diperlukan beberapa tahap sehingga hasil penyalinan *Rational Suite Project* dapat dijalankan di Rational Robot [Moving RSP]. Prosesnya adalah sebagai berikut:
 - a. Buka IBM Rational Administrator, dan pilih menu *File* → *New Project*
 - b. Masukkan nama *project* dan lokasi penyimpanannya
 - c. Tandai *Configure Project Now* dan pilih *Finish*. Seperti yang terlihat pada gambar berikut:



Gambar 4.21 Pembuatan *Rational Suite Project* Baru

Sumber: *Moving Rational Test Projects: Tips and Tricks*

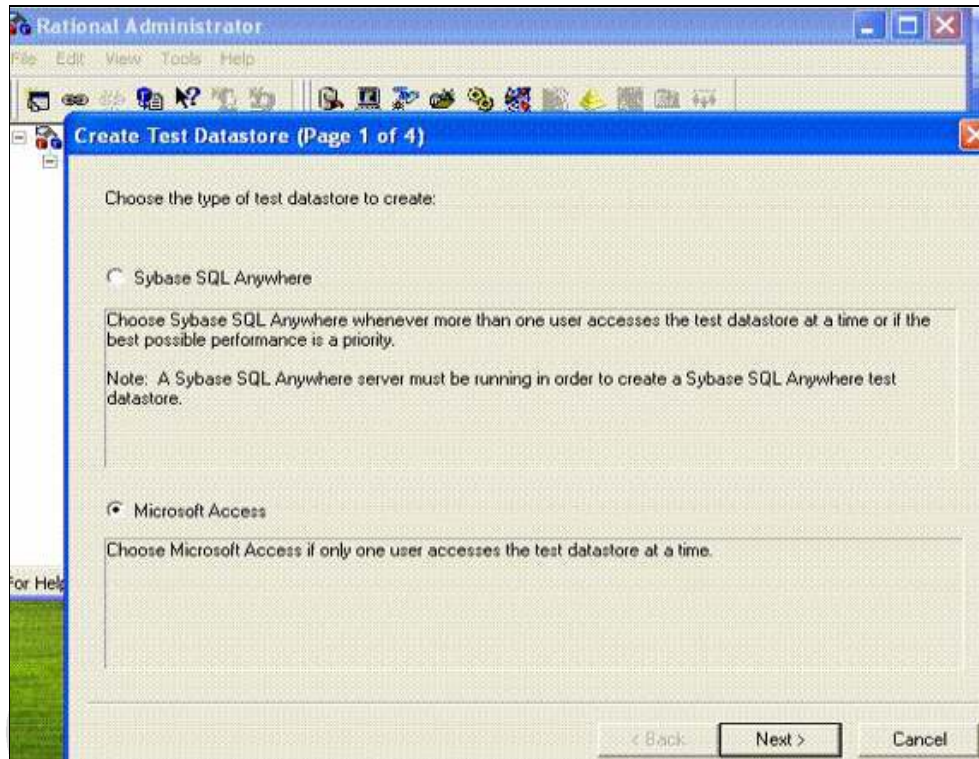
- d. Pilih *Create Test Assets* seperti yang ditunjukkan pada gambar berikut:



Gambar 4.22 Pembuatan Test Assets

Sumber: *Moving Rational Test Projects: Tips and Tricks*

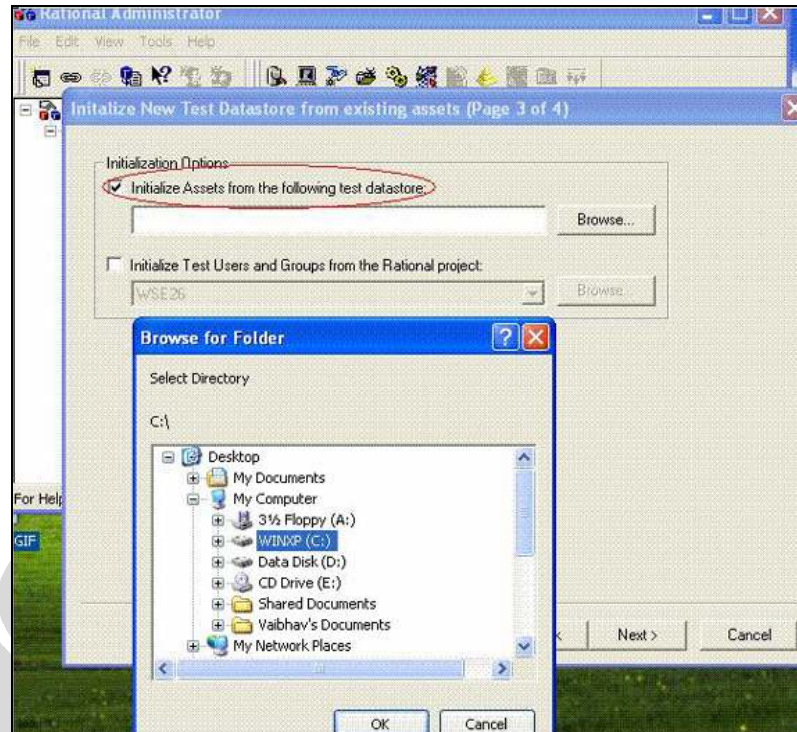
- e. Pilih *Test Datastore* seperti yang dicontohkan pada gambar berikut:



Gambar 4.23 Memilih Jenis *Test Datastore*

Sumber: *Moving Rational Test Projects: Tips and Tricks*

- f. Lakukan *browse* hingga ke direktori *test datastore*. Sebagai *default*-nya, *path* direktori yang tersedia sudah mengacu ke folder *TestDatastore* di dalam *project* baru yang telah dibuat. Gunakan saja nilai *default* ini kecuali jika terdapat alasan tertentu untuk mengubah *path* tersebut
- g. Pilih *Initialization Options* untuk menginisialisasi aset-aset dalam sebuah *datastore*. Lakukan *browse* hingga ke *folder datastore* dari *Rational Suite Project* hasil penyalinan. *Folder TestDatastore* biasanya terletak pada level *<path>Rational Project\TestDatastore*. Lakukan *browse* hingga *folder TestDatastore* dan pilih *folder* tersebut.



Gambar 4.24 Browsing Hingga ke Datastore

Sumber: *Moving Rational Test Projects: Tips and Tricks*

Sekali selesai, *Rational Suite Project* yang baru akan terinisialisasi dengan semua *test asset* yang dibuat pada *Rational Suite Project* yang lama. Semua aset telah ditransfer ke *project* baru tersebut. Persyaratan dalam menggunakan teknik ini untuk penyalinan *.rsp* adalah bahwa harus memiliki akses ke seluruh lingkungan *Rational Project*. Apabila teknik ini tidak berhasil, disarankan untuk membersihkan *folder* target sebelum melakukan usaha penyalinan lagi.

BAB 5

HASIL EKSPERIMEN DAN ANALISIS

Bab ini menjelaskan hasil eksperimen pengujian terhadap aplikasi E-Paspor. Seperti yang telah dijelaskan pada bab-bab sebelumnya, bahwa hasil pengujian Rational Robot dan Rational TestManager berupa *log* hasil pengujian. *Log* hasil pengujian beserta penjelasannya dijabarkan dalam subbab-subbab berikut. Selain *log* hasil pengujian, bab ini juga menjelaskan analisis terhadap hasil-hasil pengujian tersebut.

5.1 Log Hasil Eksperimen Pada E-Paspor

Subbab ini menjelaskan dan memperlihatkan *log-log* hasil pengujian pada aplikasi E-Paspor berdasarkan alur proses dari penerimaan permohonan hingga penyerahan dokumen.

5.1.1 Log hasil pengujian pada alur proses permohonan paspor baru

Seperti yang telah ditunjukkan pada tabel 4.3, alur kerja permohonan paspor baru terdiri dari 15 *test script*. Selain *playback* secara individual untuk masing-masing *test script*, proses *playback* untuk ke-15 *test script* ini juga menggunakan perintah *callscript* seperti yang dijelaskan pada bab 4. Dengan menggunakan *record datapool* sebanyak 5 *record*, hasil *playback* tersebut diperlihatkan oleh gambar berikut:

Event Type	Result	Date & Time	Failure Reason	Computer Name	Defects
Computer Start	Warning	6/11/2009 6:22:36 AM		PHOLETE	
Script Start (kumpulan Pasporsaja)	Pass	6/11/2009 6:22:36 AM		PHOLETE	
Call Script		6/11/2009 6:22:36 AM		PHOLETE	
Script Start (data Pasporsaja)	Pass	6/11/2009 6:22:36 AM		PHOLETE	
Call Script		6/11/2009 6:25:20 AM		PHOLETE	
Script Start (dataEntry_upload)	Pass	6/11/2009 6:25:20 AM		PHOLETE	
Call Script		6/11/2009 6:29:45 AM		PHOLETE	
Script Start (dataEntry-data Pasporsaja)	Pass	6/11/2009 6:29:45 AM		PHOLETE	
Call Script		6/11/2009 6:41:37 AM		PHOLETE	
Script Start (kasipasporsaj-anerop-Pasporsaja)	Pass	6/11/2009 6:41:37 AM		PHOLETE	
Call Script		6/11/2009 6:48:35 AM		PHOLETE	
Script Start (kasubdit-Direksi Pasporsaja)	Pass	6/11/2009 6:48:35 AM		PHOLETE	
Call Script		6/11/2009 6:51:12 AM		PHOLETE	
Script Start (kasubdit-Diplomatik Pasporsaja)	Pass	6/11/2009 6:51:12 AM		PHOLETE	
Call Script		6/11/2009 6:55:05 AM		PHOLETE	
Script Start (kasidata cetak Pasporsaja)	Pass	6/11/2009 6:55:05 AM		PHOLETE	
Call Script		6/11/2009 7:02:18 AM		PHOLETE	
Script Start (HalPengumpulDokumen Pasporsaja)	Pass	6/11/2009 7:02:18 AM		PHOLETE	
Call Script		6/11/2009 7:06:51 AM		PHOLETE	
Script Start (realisasi Pasporsaja)	Pass	6/11/2009 7:06:51 AM		PHOLETE	
Script End (kumpulan Pasporsaja)	Pass	6/11/2009 7:10:24 AM		PHOLETE	
Computer End	Warning	6/11/2009 7:10:24 AM		PHOLETE	

Gambar 5.1 Log Hasil *Playback Script* Kumpulan-Pasporsaja

Gambar 5.1 menjelaskan hasil pengujian pada alur kerja permohonan paspor baru. Dari gambar tersebut dapat dilihat bahwa hasil pengujian untuk masing-masing *test script* adalah *pass* atau sukses yang ditunjukkan dengan warna hijau. Selain itu, pada gambar juga terlihat waktu dimulainya *playback* untuk *script kumpulan-Pasporsaja*, yang sebelumnya telah dijelaskan bahwa *script* ini adalah *script* tambahan untuk memanggil semua *script* yang terlibat dalam alur proses permohonan paspor baru, dan waktu selesainya *playback*, yaitu dimulai pada 6:22:36 AM hingga 7:10:24 AM pada tanggal 11 Juni 2009. Waktu dimulainya masing-masing *script* juga dapat dilihat pada kolom **Date & Time**.

Gambar 5.1 hanya menunjukkan hasil pengujian untuk masing-masing *test script* yang dipanggil pada *script kumpulan-Pasporsaja*, yaitu sejumlah 9 *test script*. Sedangkan yang terlibat dalam alur proses permohonan paspor baru ini adalah 15 *test script* seperti yang disebutkan pada tabel 4.3. Telah dijelaskan sebelumnya, bahwa sisa *test script* yang terlibat telah dipanggil pada masing-masing *test script* dalam *script kumpulan-Pasporsaja*. Log hasil *playback*-nya juga dapat dilihat dengan meng-*expand log kumpulan-Pasporsaja* pada masing-masing *log script* seperti yang ditampilkan pada gambar 5.2.

Event Type	Result	Date & Time	Failure Reason	Computer Name	Defects
Computer Start	Warning	6/11/2009 6:22:36 AM		PHIOLETE	
Script Start (kumpulan Pasporsaja)	Pass	6/11/2009 6:22:36 AM		PHIOLETE	
Call Script		6/11/2009 6:22:36 AM		PHIOLETE	
Script Start (data Pasporsaja)	Pass	6/11/2009 6:22:36 AM		PHIOLETE	
Call Script		6/11/2009 6:22:36 AM		PHIOLETE	
Script Start (berhasil login)	Pass	6/11/2009 6:22:36 AM		PHIOLETE	
Script End (peneriksaan login)	Pass	6/11/2009 6:22:57 AM		PHIOLETE	
Playback Warning	Warning	6/11/2009 6:25:14 AM		PHIOLETE	
Uninspected Active Window	Warning	6/11/2009 6:25:17 AM		PHIOLETE	
Script End (data Pasporsaja)	Pass	6/11/2009 6:25:20 AM		PHIOLETE	
Call Script		6/11/2009 6:25:20 AM		PHIOLETE	
Script Start (dataEntry_upload)	Pass	6/11/2009 6:25:20 AM		PHIOLETE	
Call Script		6/11/2009 6:29:45 AM		PHIOLETE	
Script Start (dataEntry-data Pasporsaja)	Pass	6/11/2009 6:29:45 AM		PHIOLETE	
Call Script		6/11/2009 6:41:37 AM		PHIOLETE	
Script Start (kasipersonanep Pasporsaja)	Pass	6/11/2009 6:41:37 AM		PHIOLETE	
Call Script		6/11/2009 6:48:35 AM		PHIOLETE	
Script Start (kasubdi Dinas Pasporsaja)	Pass	6/11/2009 6:48:35 AM		PHIOLETE	
Call Script		6/11/2009 6:51:12 AM		PHIOLETE	
Script Start (kasubdi Diplomatik Pasporsaja)	Pass	6/11/2009 6:51:12 AM		PHIOLETE	
Call Script		6/11/2009 6:55:05 AM		PHIOLETE	
Script Start (kasidata cetak Pasporsaja)	Pass	6/11/2009 6:55:05 AM		PHIOLETE	
Call Script		6/11/2009 7:02:18 AM		PHIOLETE	
Script Start (statpengumpul dokumen Pasporsaja)	Pass	6/11/2009 7:02:18 AM		PHIOLETE	
Call Script		6/11/2009 7:06:51 AM		PHIOLETE	
Script Start (resahkan Pasporsaja)	Pass	6/11/2009 7:06:51 AM		PHIOLETE	
Script End (kumpulan Pasporsaja)	Pass	6/11/2009 7:10:24 AM		PHIOLETE	
Computer End	Warning	6/11/2009 7:10:24 AM		PHIOLETE	

Gambar 5.2 Log yang Di-expand

5.1.2 Log hasil pengujian pada alur proses permohonan paspor baru dan exit permit

Seperti yang telah ditunjukkan pada tabel 4.3, alur kerja permohonan paspor baru dan exit permit terdiri dari 20 *test script*. Masing-masing *test script* telah melalui proses *playback* untuk mengetahui bahwa *test script* berfungsi secara benar. Ke-20 *test script* tersebut juga telah digabungkan menjadi satu dalam *script* tambahan, yaitu *script kumpulan-PasporEP*, dengan menggunakan perintah *callscript* seperti yang telah dijelaskan pada bab 4. Log hasil pengujian yang akan ditampilkan dalam laporan Tugas Akhir ini adalah log hasil *playback script kumpulan-PasporEP* karena dengan melihat log ini saja sudah melingkupi seluruh 20 *script* tersebut. Gambar 5.3 menunjukkan log hasil *playback script kumpulan-PasporEP* dengan menggunakan *record datapool* sebanyak 5 *record*.

Event Type	Result	Date & Time	Failure R...	Computer Na...	Defects
Computer Start	Warning	6/11/2009 9:32:48 AM		PHIOLETE	
Script Start (kumpulan-PasporEP)	Pass	6/11/2009 9:32:48 AM		PHIOLETE	
Call Script	Pass	6/11/2009 9:32:48 AM		PHIOLETE	
Script Start (daftar-PasporEP)	Pass	6/11/2009 9:34:49 AM		PHIOLETE	
Call Script	Pass	6/11/2009 9:34:49 AM		PHIOLETE	
Script Start (dataEntry_upload)	Pass	6/11/2009 9:39:14 AM		PHIOLETE	
Call Script	Pass	6/11/2009 9:39:14 AM		PHIOLETE	
Script Start (dataEntry-data-PasporEP)	Pass	6/11/2009 9:39:14 AM		PHIOLETE	
Call Script	Pass	6/11/2009 9:51:59 AM		PHIOLETE	
Script Start (kaospor-emas-PasporEP)	Pass	6/11/2009 9:51:59 AM		PHIOLETE	
Call Script	Pass	6/11/2009 9:59:02 AM		PHIOLETE	
Script Start (kaospor-aspal-PasporEP)	Pass	6/11/2009 9:59:02 AM		PHIOLETE	
Call Script	Pass	6/11/2009 10:03:53 AM		PHIOLETE	
Script Start (kaosbdi-dinas-PasporEP)	Pass	6/11/2009 10:03:53 AM		PHIOLETE	
Call Script	Pass	6/11/2009 10:10:27 AM		PHIOLETE	
Script Start (kaosbdi-Diplomatik-PasporEP)	Pass	6/11/2009 10:10:27 AM		PHIOLETE	
Call Script	Pass	6/11/2009 10:20:03 AM		PHIOLETE	
Script Start (kaosdata-cetak-Paspor)	Pass	6/11/2009 10:20:03 AM		PHIOLETE	
Call Script	Pass	6/11/2009 10:27:16 AM		PHIOLETE	
Script Start (stapok-cetak)	Pass	6/11/2009 10:27:16 AM		PHIOLETE	
Call Script	Pass	6/11/2009 10:31:10 AM		PHIOLETE	
Script Start (stapungunpudbakuram-Pas...)	Pass	6/11/2009 10:31:10 AM		PHIOLETE	
Call Script	Pass	6/11/2009 10:35:47 AM		PHIOLETE	
Script Start (sarafkan-PasporEP)	Pass	6/11/2009 10:35:47 AM		PHIOLETE	
Script End (kumpulan-PasporEP)	Pass	6/11/2009 10:39:35 AM		PHIOLETE	
Computer End	Warning	6/11/2009 10:39:35 AM		PHIOLETE	

Gambar 5.3 Log Hasil *Playback Script* Kumpulan-PasporEP

Gambar 5.3 menjelaskan hasil pengujian pada alur proses permohonan paspor baru dan exit permit. Seperti *log* hasil *playback script* kumpulan-PasporSaja, *log* hasil *playback script* kumpulan-PasporEP juga menunjukkan bahwa aplikasi E-Paspor berjalan dengan baik pada alur proses permohonan paspor baru dan exit permit. Hal ini ditunjukkan oleh tanda *pass* dan warna hijau pada masing-masing *test script* yang terlibat dalam alur proses tersebut. Waktu dimulainya *playback* adalah 9:32:48 AM dan berakhir pada 10:39:35 AM, atau *playback* berjalan selama 1 jam 6 menit 47 detik. Waktu yang ditempuh oleh *playback* tersebut lebih lama dibandingkan dengan *playback* pada *script* kumpulan-PasporSaja, yaitu hanya 47 menit dan 48 detik.

5.1.3 Log hasil pengujian pada alur kerja permohonan paspor baru, exit permit, dan rekomendasi visa

Jumlah *test script* yang digunakan untuk menjalankan pengujian secara menyeluruh pada alur proses permohonan paspor baru, exit permit, dan rekomendasi visa adalah 23 buah *script*. Seperti penjelasan sebelumnya, bahwa ke-23 *script* tersebut telah digabungkan dalam satu *script* tambahan dengan nama **kumpulan**. Kemudian dilakukan *playback* terhadap *script* tersebut dengan

Universitas Indonesia

memakai *record datapool* sejumlah 5 *record* sehingga menghasilkan *log* hasil *playback* yang ditunjukkan pada gambar berikut:

Event Type	Result	Date & Time	Failure R...	Computer Na...	Defec
Computer Start	Warning	6/11/2009 12:53:48 PM		PHIOLETE	
Script Start (kumpulan)	Pass	6/11/2009 12:53:48 PM		PHIOLETE	
Call Script		6/11/2009 12:53:48 PM		PHIOLETE	
Script Start (daftar)	Pass	6/11/2009 12:53:48 PM		PHIOLETE	
Call Script		6/11/2009 12:55:52 PM		PHIOLETE	
Script Start (dataEntry_upload)	Pass	6/11/2009 12:55:52 PM		PHIOLETE	
Call Script		6/11/2009 1:00:20 PM		PHIOLETE	
Script Start (dataEntry-data)	Pass	6/11/2009 1:00:20 PM		PHIOLETE	
Call Script		6/11/2009 1:12:30 PM		PHIOLETE	
Script Start (kasipaspor-amerop)	Pass	6/11/2009 1:12:30 PM		PHIOLETE	
Call Script		6/11/2009 1:21:17 PM		PHIOLETE	
Script Start (kasipaspor)	Pass	6/11/2009 1:21:18 PM		PHIOLETE	
Call Script		6/11/2009 1:27:23 PM		PHIOLETE	
Script Start (kasubdi-diras)	Pass	6/11/2009 1:27:23 PM		PHIOLETE	
Call Script		6/11/2009 1:35:23 PM		PHIOLETE	
Script Start (kasubdi-Diplonak)	Pass	6/11/2009 1:35:23 PM		PHIOLETE	
Call Script		6/11/2009 1:47:33 PM		PHIOLETE	
Script Start (kasidata-cetak-Paspor)	Pass	6/11/2009 1:47:33 PM		PHIOLETE	
Call Script		6/11/2009 1:54:46 PM		PHIOLETE	
Script Start (stafp-cetak)	Pass	6/11/2009 1:54:46 PM		PHIOLETE	
Call Script		6/11/2009 1:59:40 PM		PHIOLETE	
Script Start (stafp-cetak)	Pass	6/11/2009 1:59:40 PM		PHIOLETE	
Call Script		6/11/2009 2:00:08 PM		PHIOLETE	
Script Start (stafpempubdikuman)	Pass	6/11/2009 2:00:08 PM		PHIOLETE	
Call Script		6/11/2009 2:04:48 PM		PHIOLETE	
Script Start (zembali)	Pass	6/11/2009 2:04:48 PM		PHIOLETE	
Script End (kumpulan)	Pass	6/11/2009 2:08:51 PM		PHIOLETE	
Computer End	Warning	6/11/2009 2:08:51 PM		PHIOLETE	

Gambar 5.4 Log Hasil *Playback Script* Kumpulan

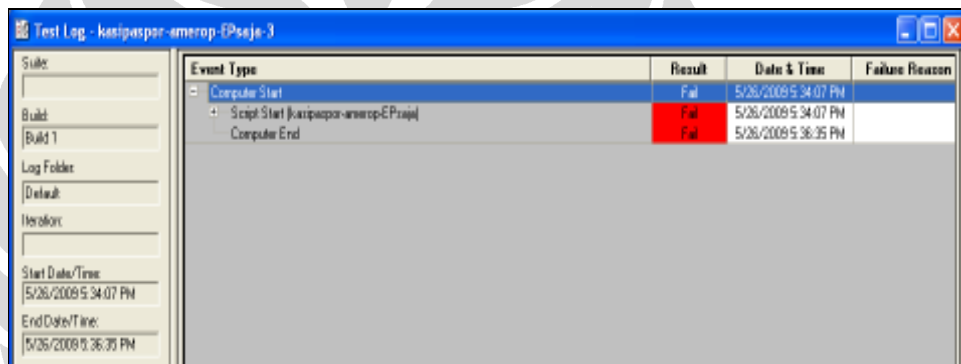
Gambar 5.4 menunjukkan bahwa aplikasi E-Paspor berjalan dengan baik pada alur proses permohonan paspor baru, exit permit, dan rekomendasi visa. Hal ini ditunjukkan dengan tanda *pass* dan warna hijau pada masing-masing log *script*. Proses *playback* dilakukan mulai 12:53:48 PM hingga 2:08:51 atau selama 1 jam 15 menit 3 detik. Proses *playback* untuk *script* **kumpulan** ini lebih lama bila dibandingkan dengan proses *playback* untuk *script* **kumpulan-PasporSaja** dan **kumpulan-PasporEP** yang masing-masing hanya selama 47 menit 48 detik dan 1 jam 6 menit 47 detik.

5.1.4 Log Hasil Pengujian Pada Alur Kerja Permohonan Exit Permit

Tidak ada *script* tambahan yang mengumpulkan *script-script* yang terlibat dalam alur proses permohonan exit permit. Hal ini dikarenakan pada saat dilakukan *playback* individual pada masing-masing *script* yang terdaftar dalam tabel 4.3, saat tiba pada *script* **kasipaspor-amerop-EPsaja**, ditemukan suatu kerusakan

pada sistem E-Paspor, khususnya pada fungsi memberikan keputusan kelengkapan dokumen oleh kasi paspor. Untuk *script-script* yang dijalankan sebelum *script kasipaspor-amerop-EPsaja*, tidak ada permasalahan yang terjadi. Semua *log* hasil *playback* pada masing-masing *script* tersebut menunjukkan tanda *pass* dan warna hijau yang menandakan bahwa fungsi-fungsi E-Paspor pada alur proses sebelum penanganan permohonan oleh kasi paspor telah berjalan dengan baik.

Adapun kerusakan yang terjadi pada E-Paspor ditunjukkan pada *log* hasil *playback script kasipaspor-amerop-EPsaja* berikut:



Event Type	Result	Date & Time	Failure Reason
Computer Start	Fail	5/26/2009 5:34:07 PM	
Script Start (kasipaspor-amerop-EPsaja)	Fail	5/26/2009 5:34:07 PM	
Computer End	Fail	5/26/2009 5:36:35 PM	

Gambar 5.5 Log Hasil *Playback Script Kasipaspor-amerop-EPsaja*

Berdasarkan gambar 5.5 terlihat bahwa ringkasan *log* adalah gagal (*fail*) yang ditunjukkan oleh warna merah dan tanda *fail*. Apabila *log* tersebut di-*expand*, maka akan terlihat lebih spesifik apa yang menyebabkan *script* menjadi gagal. *Log* yang sudah di-*expand* ditunjukkan oleh gambar 5.6

Event Type	Result	Date & Time	Failure Reason
Computer Start	Fail	5/26/2009 5:34:07 PM	
Script Start (kasipaspor-amerop-EPsaja)	Fail	5/26/2009 5:34:07 PM	
Call Script		5/26/2009 5:34:07 PM	
Script Start (kasipaspor-amerop-login)	Pass	5/26/2009 5:34:07 PM	
Script End (kasipaspor-amerop-login)	Pass	5/26/2009 5:34:26 PM	
Verification Point (Object Properties - Object Properties)	Pass	5/26/2009 5:34:32 PM	
Playback Warning	Warning	5/26/2009 5:35:02 PM	
Log Message (baseline value: 'Maureen Rogge' = Actual value: 'Maureen Rogge')	Pass	5/26/2009 5:35:02 PM	
Verification Point (Object Properties - Object Properties)	Pass	5/26/2009 5:35:36 PM	
Playback Warning	Warning	5/26/2009 5:36:06 PM	
Log Message (baseline value: 'Pedro Sprankle' != Actual value: 'Maureen Rogge')	Fail	5/26/2009 5:36:06 PM	
Script End (kasipaspor-amerop-EPsaja)	Fail	5/26/2009 5:36:35 PM	
Computer End	Fail	5/26/2009 5:36:35 PM	

Gambar 5.6 Log Hasil *Playback Script* Kasipaspor-amerop-EPsaja

Gambar 5.6 memperlihatkan bahwa *script* sempat mengalami sukses (*pass*) yang ditunjukkan oleh warna hijau. Kegagalan *script* dimulai pada *verification point* yang dibuat secara manual dengan menggunakan perintah **if...else**, **sqlLogMessage**, dan **sqlGetPropertyAsString** seperti yang dijelaskan pada bab 4, yaitu pada saat membandingkan nilai *baseline* yang berasal dari *datapool* dan nilai aktual yang ditangkap pada GUI E-Paspor pada saat *playback*. Jika dilihat lebih dalam, *verification point* yang gagal membandingkan keduanya adalah *verification point* yang kedua, yaitu yang ditunjuk oleh garis bermata panah.

Verification point ini menunjukkan bahwa nilai *baseline*, yaitu Pedro Sprankle, tidak sama dengan nilai aktual, yaitu Maureen Rogge. Hal ini berarti bahwa permohonan oleh Maureen Rogge belum diproses dengan sukses oleh kasi paspor karena pada *verification point* manual yang pertama, Maureen Rogge sudah dibandingkan dan menghasilkan *pass* karena sama dengan nilai *baseline*-nya.

Dengan melihat *log* seperti ini, penulis kemudian menelusuri GUI E-Paspor, khususnya GUI untuk kasi paspor. Hasilnya adalah terdapat satu isian pada *edit box* yang *mandatory* yang wajib diisi, tetapi tidak dapat diisi karena status *property* pada *edit box* tersebut *disabled*. Tampilan GUI kasi paspor diperlihatkan pada gambar 5.7.

The screenshot shows the 'DEPARTEMEN LUAR NEGERI REPUBLIK INDONESIA' interface. The main section is titled 'Perencanaan Perencanaan' and 'Perencanaan Paspor Perencanaan'. Below this, there's a section for 'Data Pribadi Pemohon (No.Reg: D00006/26/05/09)'. The form contains several input fields: 'Nama Lengkap' (filled with 'Maurin Sogor'), 'Nama Internet' (filled with 'PT. MAURIN'), 'Nama Paspor' (filled with '000012'), 'Tanggal Urah' (filled with '05/05/09'), 'Jenis Paspor' (filled with 'Luar-Negeri'), 'Jenis Kelamin' (filled with 'Laki-Laki'), 'Jenis Pekerjaan' (filled with 'Pegawai Negeri'), 'Jenis Pendidikan' (filled with 'Diploma Diklat'), 'Jenis Pekerjaan' (filled with 'Pegawai Negeri'), 'Jenis Pendidikan' (filled with 'Diploma Diklat'), 'Jenis Pekerjaan' (filled with 'Pegawai Negeri'), 'Jenis Pendidikan' (filled with 'Diploma Diklat'). A red arrow points to a text input field labeled 'Nama status rumah (Rumah/Hotel) yang sesuai dalam kategori: (*)' which is disabled. Below the form are buttons for 'Perencanaan Paspor Baru' and 'Perencanaan Paspor Baru'.

Gambar 5.7 Tampilan GUI Kasi Paspor

Edit box yang ditunjuk oleh garis bermata panah pada gambar 5.7 menunjukkan *edit box* yang statusnya *disabled* sehingga tidak dapat dimasukkan nilai apapun. Padahal, isian pada *edit box* ini adalah *mandatory* dan harus diisi. Jika tidak diisi, maka proses permohonan exit permit tidak dapat dilanjutkan.

5.1.5 Hasil Eksperimen

Subbab ini menjelaskan hasil eksperimen pengujian pada aplikasi E-Paspor untuk ketiga alur proses yang telah dijelaskan. Hasil eksperimen ini sendiri berupa durasi waktu pengujian selama *playback script* untuk sejumlah *record datapool*. Tabel berikut menunjukkan hasil eksperimen tersebut.

Tabel 5.1 Hasil Eksperimen

	<i>5 record datapool</i>	<i>10 record datapool</i>	<i>20 record datapool</i>
kumpulan-pasporSaja	47 menit 48 detik	1 jam 32 menit 39 detik	2 jam 59 menit 40 detik
kumpulan-pasporEP	1 jam 6 menit 47 detik	2 jam 11 menit 35 detik	4 jam 29 menit 3 detik
kumpulan	1 jam 15 menit 3 detik	2 jam 21 menit 55 detik	4 jam 41 menit 39 detik

Tabel 5.1 memperlihatkan hasil eksperimen pada ketiga alur proses pelayanan E-Paspor untuk beberapa jumlah *record datapool* yang digunakan dalam pengujian. Waktu atau durasi yang dibutuhkan untuk masing-masing alur proses dari pendaftaran hingga penyerahan dokumen permohonan terlihat semakin besar (semakin lama) selaras dengan peningkatan jumlah *record datapool* yang digunakan dalam eksperimen. Begitu juga dengan peningkatan durasi yang terjadi pada setiap kolom yang mengalami peningkatan seiring bertambahnya *script* yang dijalankan.

5.2 Analisis Hasil Pengujian Pada E-Paspor

Subbab ini membahas analisis hasil eksperimen pengujian E-Paspor yang menggunakan Rational Robot dan Rational TestManager yang berupa *test log*. Berdasarkan *test log* yang dihasilkan oleh *playback script*, berikut analisis terhadapnya.

1. Semakin banyak *script* yang dipanggil dalam suatu *script* tambahan, maka semakin lama *playback* berlangsung. Hal ini terbukti dengan waktu yang ditempuh oleh *script kumpulan* yang memanggil sejumlah 23 *script* lebih lama dari waktu yang ditempuh oleh *script kumpulan-PasporSaja* dan *kumpulan-PasporEP* yang masing-masing memanggil sejumlah 20 dan 15 *script*. Begitu juga dengan waktu yang ditempuh oleh *script kumpulan-PasporEP* lebih lama dibandingkan dengan waktu yang ditempuh oleh *script kumpulan-PasporSaja*.
2. Semakin banyak *record datapool* yang digunakan untuk pengujian, semakin lama juga waktu atau durasi yang dibutuhkan untuk menyelesaikan pengujian tersebut.

3. Pembuatan *script* yang modular membantu mempermudah pengujian karena mendukung sifat *reusability* sehingga menghemat baris kode pada *script*. Pernyataan ini didukung oleh *log-log* hasil *playback* yang memperlihatkan sejumlah *script* yang digunakan pada beberapa alur proses yang berbeda, seperti *script penerimaan-login, dataEntry-login, dataEntry_upload, kasipasporamerop-login, kasipasporaspasaf-login, LoginKasubdit*, dan lain sebagainya, yang digunakan pada semua alur proses (lihat tabel 4.3).
4. Sebagian besar fungsi-fungsi dalam E-Paspor berjalan dengan baik, kecuali fungsi memberikan keputusan kelengkapan oleh kasi paspor pada alur proses permohonan exit permit.
5. Fungsi-fungsi aplikasi pada alur proses permohonan paspor baru dan exit permit merupakan pengembangan dari fungsi-fungsi pada alur proses permohonan paspor baru saja. Sedangkan fungsi-fungsi pada alur proses permohonan paspor baru, exit permit, dan rekomendasi visa adalah hasil pengembangan dari fungsi-fungsi aplikasi E-Paspor pada alur proses permohonan paspor baru dan alur proses permohonan paspor baru dan exit permit. Hal ini menunjukkan bahwa Rational Robot dan Rational TestManager mendukung *regression testing*. Penjelasannya adalah bahwa pengujian pada alur proses permohonan paspor baru dapat dilakukan kembali secara otomatis hanya dengan melakukan *playback* setelah pengujian terhadap aplikasi E-Paspor pada alur proses permohonan paspor baru dan exit permit dilakukan. Begitu pula dengan *test script* yang dihasilkan dalam pengujian pada alur proses permohonan paspor baru dan exit permit tersebut, dapat di-*playback* kembali untuk memastikan bahwa fungsi-fungsi pada alur tersebut tidak terpengaruh oleh penambahan fungsi permohonan rekomendasi visa.