MOVING AWAY FROM SPREADSHEET ENGINEERING WITH ECLIPSE TECHNOLOGY
VIEW FROM MY DESK...

Christophe Bouhier
17 years working in ‘coms’ domain

netxforge.com
GOAL FOR TODAY

INSPIRE!

Eclipse feels like Lego again! Little bricks one can put together.
5.2.4 Identify the key resources that determine the network element capacity, these are usually defined by the vendor.

5.2.5 Determine User Service Profile

In this step, the resource utilization information is used to extract a typical service profile. The service usage profile can be extracted for resources which provide information on the service usage and the number of users, using the service.

Example: The user service profile for the voice service is determined in the following way. An MSC-Server provides resource usage for VLR–Attached Subscribers and total aggregated traffic through the MSC-Server. The user service profile for voice is derived by dividing the traffic by the VLR–Attached subscribers. Note, in this case no distinction is made on derived services like Prepaid Voice or Business Voice. For this additional resources could be used from other functions like the HLR or the prepaid platform.

The resource which will be chosen to calculate the user service profile should be chosen with great care. Multiple usage measurements can be collected from the elements in the service chain; however usage measurements are subject to the effect of varying busy hour, statistical multiplexing and perhaps also the type of interface and might contain protocol overheads.

Example: The traffic volumes for voice are measured on the transcoders but also on the MSC Servers. When adding the traffic volumes for each of the transcoders for each of their respective busy hour, the sum will be higher than measured on the MSC Server at its busiest hour.

5.2.6 Create Traffic Light Service Utilization report

In this step, the total chain of Network Element Instances and corresponding traffic lights are consolidated into a single report. In addition to the report, the status of the chain is stored in the Central Capacity Management data repository for future reference.

The Service Utilization can be reported as a chain, see figure below.

Or in a hierarchical fashion (according the hierarchy of the service), see figure below.
ENRICHMENT OF NETWORK METRICS

- **CAP Calc**
- **TOL Calc**
- **TOL Calc**
- **TOL Calc**

- Marker **Start:GREEN**
- Marker **UP:AMBER**
- Marker **DOWN:RED**
- Marker **DOWN:AMBER**
- Marker **DOWN:GREEN**

- **CAP**
- **RED**
- **AMBER**
- **YELLOW**
SPREADSHEET ENGINEERING

Capacity Metric data processed per single flow (thousands of flows exist!)

Multiple manual actions in processing steps.

Aggregation on higher-level order secondary laborious task.

Auditing, quality checks difficult. (Formula’s embedded in large spreadsheets).

In other words… typical MS Office automation…
OUR APPROACH

- Metric data processed per 1000th of flows
- No manual Actions during data processing
- Metric data Calculations
- Auditing, quality checks easy, through shared data.
- In other words… typical BI implementation but…. 
UI => ISCREEN FRAMEWORK

Capabilities

- Flexible / Sexy presentation, (No IEditorPart)
- Navigation to sub-screens.
- Powerful central action handling / Selection / Undo/Redo
- Support for Roles, Users, Activities (UI hiding), Permissions (Editing, Read-Only)
- IDE, Navigator with Workspace selection, Workspace reset (when role changed).
- External UI Refresh support (CDO Invalidations backed).
- Best practice Widgets implementations i.e a background loading TableViewer for very large and long loading tables.

API

No, but with some help IScreen’s could be a standalone framework. Some initial steps taken to abstract away the functionality.

:DEMO => ‘IScreen’
QUARTZ JOBS & CDO

Integration of Quartz as a service
- Quarts has own Job definition.
- Storing Job data in CDO
- UI for managing Jobs
- Job progress

:DEMO => ‘Jobs’
XTEXT INTEGRATION
(EXECUTION LOGIC)

- Storing of Xtext content in CDO
- Embedding Xtext in JFace forms. (TODO: mention the source).
- CDO Enabled Global Scope Provider.

Telco Equipment => Real
Telco Equipment => Modeled

:DEMO => ‘NetXScript’

```plaintext
// Capacity
this CAP = PARAM STP_MSU_CAPACITY;

// Utilization
this UTILIZATION = this METRIC AVG 15 / this CAP;
```
GIS INTEGRATION
(GEOFF/OPEN LAYERS)

https://github.com/erdalkaraca/geoff
LESSONS LEARNED:

• Eclipse Technology development rate is very high!

• Leaky abstraction theory is true in this case.

• Modeling technology contributions are targeted towards SW Engineering, In our case, we use modeling technology at the heart of the application. Some Attention to this mindset would be great.
  • Live Model Migration
  • Fixture handling (Import/Export various formats).
  • Object Revision handling is heavy (But can be partly turned off).

• FUN:
  • HSSF => Horrible Spreadsheet Format
  • XSSF => eXtremely Horrible Spreadsheet Format (??)
Come by on:

THANK YOU
:INDEPTH =>
DECISIONS

UI

Web based or RCP?

With to have a web-based solution (RAP, GWT, JSP etc..) considered to be to challenging/risky
In combination with requirement for Xtext and other aspects like diagraming.

4.x not advised to be used in 2011. Many learnings building RCP for 3.x, so decided to stick On 3.x. Evaluated code generation. (EEF), GMT. Considered not suitable or not understood well enough for the purpose. XWT not suitable.

Challenge with RCP: The standard “navigator/editor” paradigm not suitable. (Not visually satisfying for the UI intended).
DESKTOPS

Modeling, Persistence etc..

No real evaluation for persistence and object relational framework. Some experience with Teneo, Single user client used in a prototype. Settled for CDO, but deep dive without knowing what to expect. Mitigated risk, by hiring modeling specialist (Martin Taal).

Requirement for processing, presentation, storage etc… of large volumes of volatile “live” data has been a continuous source of concern (and still is).

DBStore for CDO, initially HibernateStore, but switched to DBStore (which we discovered is more responsive).
THE COMPONENTS 1(2)

• Use Eclipse Technologies! (Based on availability in 3.7[TODO, name])
• Use Other Open Source Licensed Components.

• OSGI (Equinox)
• Platform
  • 3.x Workbench
  • IDE, Navigator, Console, Properties
• Modeling technologies:
  • EMF => Foundation for modeling
  • CDO => Persistence, Subscription Management, Multi User, Change Notification etc..
  • Xtext => Scripting formula editing (2.x)
  • Xpand/Xtend (1.0) => Model transform to HTML etc..
• RCP, Jface Viewers, Databinding, SWT, Nebula, SWTChart
• GEF, Draw2d
THE COMPONENTS 2(2)

• Use Other Open Source Licensed Components. Many leveraged by Eclipse itself.

• Quartz Job Scheduler => Scheduled background job processing
• Dependency Injection (Guice) => Modularized approach
• Apache technologies
  • Math => Used by scripting calculations
  • POI (MS formats). => Import/Export of MS Excel content
  • Logging, Commons etc…
  • XML Technologies

…. But challenging various integration points identified, and to solve