experience success

simplify experience | harness data | stay ahead | be efficient

Practical Lessons in Developing an Enterprise-Wide Semantic Application

Craig Hanson

Semtech 2012

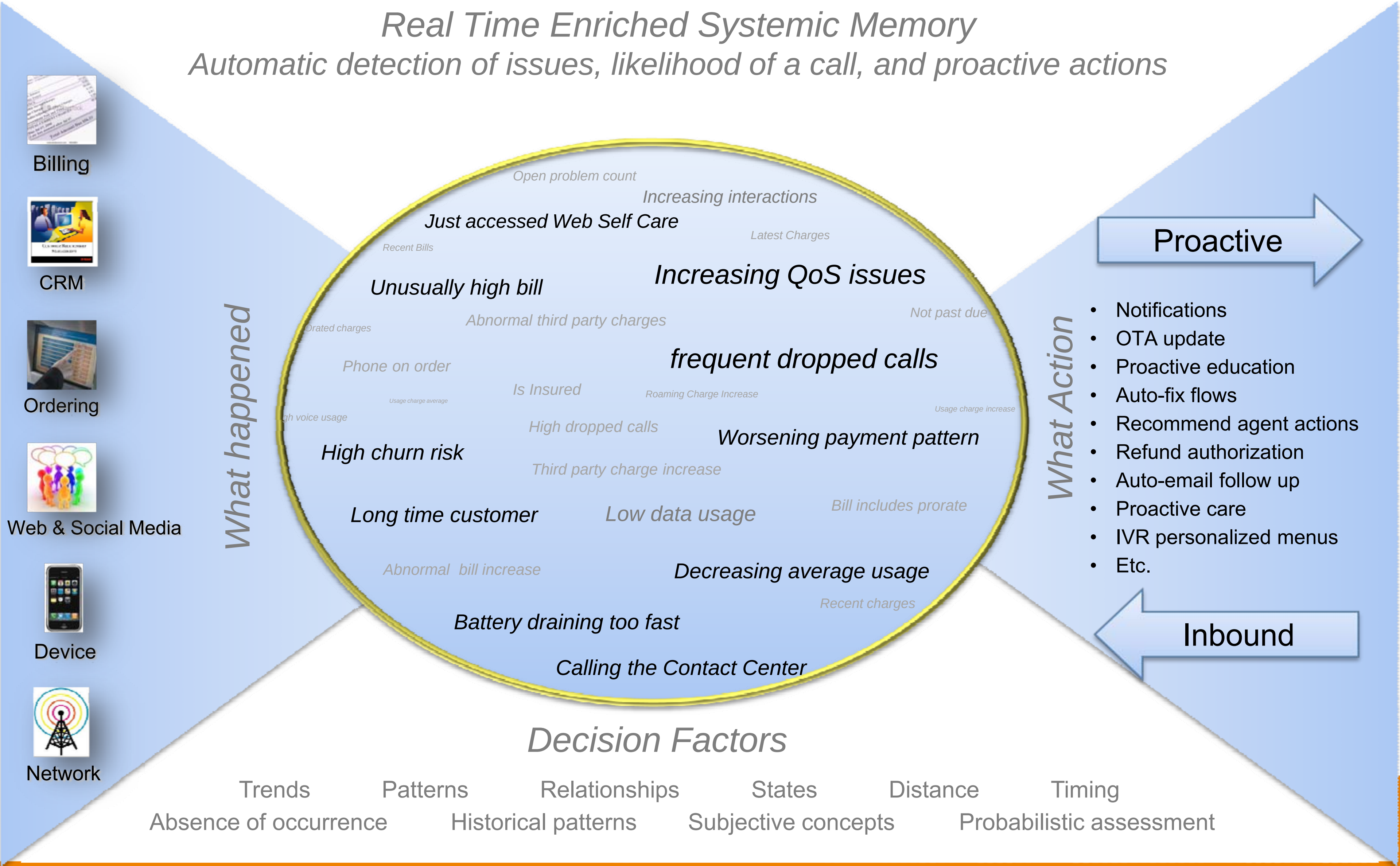
Twitter - @craighanson1
Craig.Hanson@amdocs.com

amdocs

embrace challenge experience success

Amdocs Proactive Insight

Real-time Events ➔ Contextual Decisions ➔ Actions



Real Time Enriched Systemic Memory

Automatic detection of issues, likelihood of a call, and proactive actions

Billing

CRM

Ordering

Web & Social Media

Device

Network

What happened

Open problem count

Increasing interactions

Just accessed Web Self Care

Latest Charges

Recent Bills

Unusually high bill

Increasing QoS issues

Not past due

orated charges

Abnormal third party charges

frequent dropped calls

Phone on order

Is Insured

Roaming Charge Increase

Usage charge average

Usage charge increase

gh voice usage

High dropped calls

Worsening payment pattern

High churn risk

Third party charge increase

Long time customer

Low data usage

Bill includes prorate

Abnormal bill increase

Decreasing average usage

Recent charges

Battery draining too fast

Calling the Contact Center

What Action

Proactive

- Notifications
- OTA update
- Proactive education
- Auto-fix flows
- Recommend agent actions
- Refund authorization
- Auto-email follow up
- Proactive care
- IVR personalized menus
- Etc.

Inbound

Decision Factors

Trends Patterns Relationships States Distance Timing

Absence of occurrence Historical patterns Subjective concepts Probabilistic assessment

Communicate Business Value

Practical lesson #1

Speak the language of the business

Communicate Business Value

Practical lesson

Ontology

OWL

RDF

RETE

RDFS

Triples

Axiom

RIF

Inference

URI

Provenance

Semantic

Jena

Upper ontology

SPARQL

SKOS

What we want to say

- 👍 Save dollars
- 👍 Complete faster
- 👍 Increase sales
- 👍 Increase customer satisfaction

What we need to say

Communicate Business Value

Practical lesson #1

Technology	Feature (Technical)	Benefit	Business Value
RDF	Flexible data model	Rapid change	Changes take < 2 weeks
RDF	Flexible data model	No data conversion	Saves $500k per release
RDF	Linked data	Holistic view	5 new application possible
RDF	Linked Open Data	Use external data	Completed in 3 months with 5x content
RDF	Meaning is in the data	Sharing data	3 new applications leverage the data
RDF	Self describing data	Sharing data	Save 40% when developing new application
Inference Engine	Derived high level concepts	Analysts ready data	Saves 6MM development
Inference Engine	Auto categorization	Analysts ready data	Available real time vs. days
Inference Engine	Model and rules	Changes by Business people	Changes made in days

Enterprises are Closed Minded

Practical lesson #2

Use Open World Assumption
only on open world problems

Enterprises are Closed Minded

Practical lesson #2

- Truth of a statement is independent of whether or not it is *known*

Open World Assumption

Is it possible that Patrick and Craig know each other?

- Any statement that is not known to be true is false

Closed World Assumption

Customers payment processed

Phone activated

Technician dispatched

Money transferred

Don’t Be a Semantic Purist

Practical lesson #3

Use the semantic tools to solve a problem, bring unique business or technical value

Don’t Be a Semantic Purist

Practical lesson #3

- We use RDF to enable flexible, rapidly deployed changes done by people that understand the business
- We choose not to use OWL reasoners

Transmission Audio Ultimate

Audio Laboratory BP1 Dual Mono Power Amp

Audio Laboratory BC1 Pre Amp

Twelve 500 Watt speakers

- Fourty 15 inch subwoofers
- Twenty Four 8 inch woofers
- Ribbon technology for high – mid range

Your combination will be different

Performance, Performance, Performance

Practical lesson #4

Make Performance a Priority

Performance, Performance, Performance

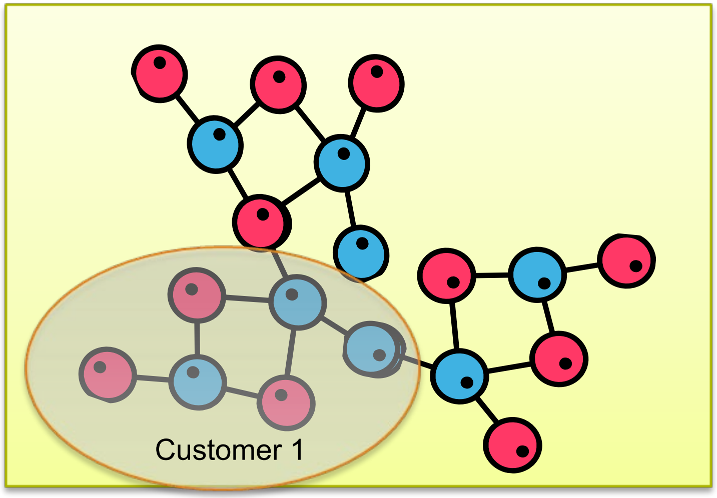
Practical lesson #4

- Practice Software Performance Engineering (SPE)
  - Design for the performance you need at scale.
- Budget based performance
- Set Architectural Tenets based on scale and performance
- Establish performance test as a part of development
  - Examine both throughput and resource demand
- Work directly with the vendors
- Purchase a real database server with plenty of memory

Performance, Performance, Performance

Practical lesson #4

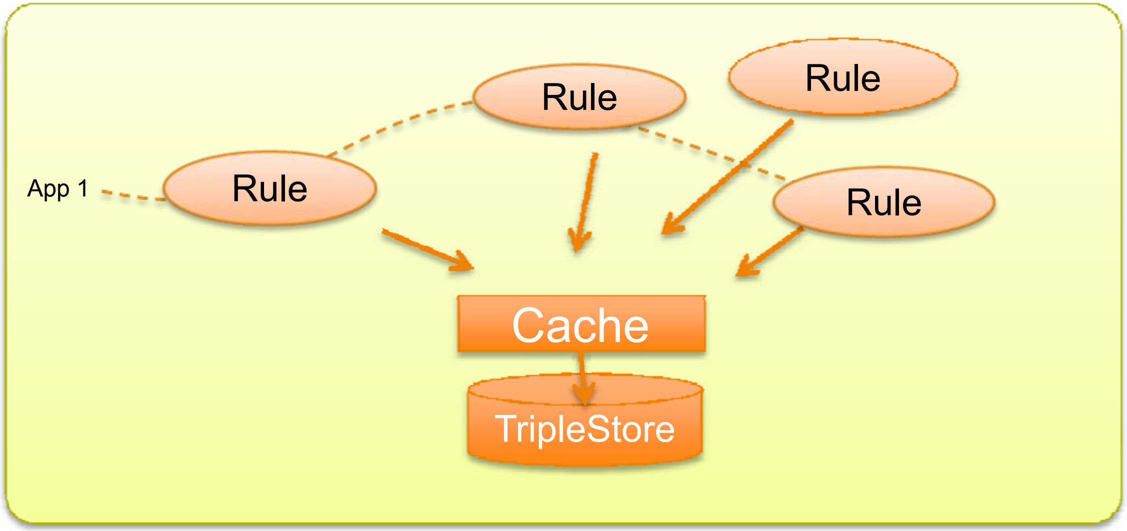
- Off the shelf inference engines
  - Computational overhead not appropriate for real time system
  - Rules trigger other rules – explosion of rule executions
    - Well known problem of forward chaining rule engines
    - Limit our language
  - Rete setup
    - At scale constructing the rete network too great
- Deal with scope of inference
  - Fire the rules for only one customer



Performance, Performance, Performance

Practical lesson #4

- Database access pattern
  - We specify our application in separate sets of rules
  - Rules are self contained performing their own queries
  - Applications string together their unique sequence of rules
  - The engine chains the rules
  - The sheer number of queries are chatty



Query and work from cache

or

Create custom super query

Performance, Performance, Performance

Practical lesson #4

- Duplicate triples
  - RDF doesn’t define it, but it’s a reality
    - You have to do it yourself
  - The functionality in the Databases comes at a cost
- Strings
  - Control the use of strings
    - Use them for readability not for computations
    - Don’t overload meaning in strings

Express Yourself (Know your Language)

Practical lesson #5

Know what the language is good at and what it is not

Express Yourself (Know your Language)

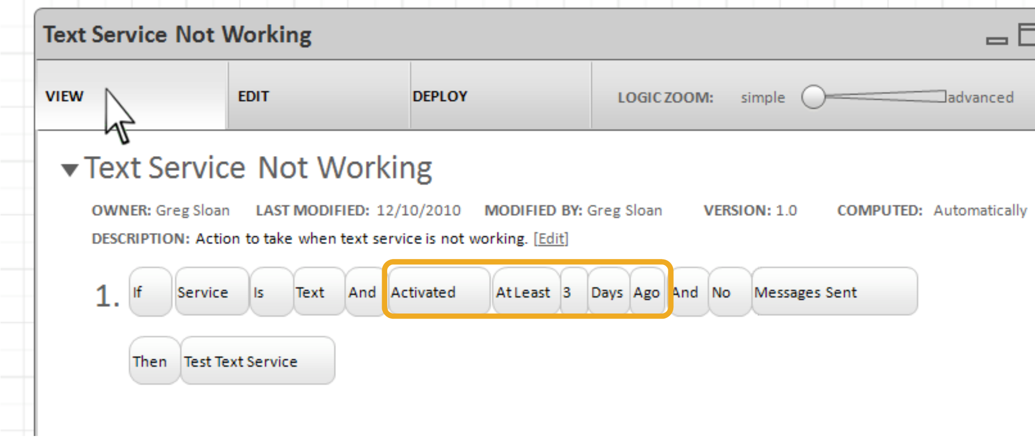
Practical lesson #5

- The modeling languages (RDF, RDFS,…)
  - Great for managing data
  - Linked Data
  - Describe any concept
    - Things that are alike
    - Things that are not alike
  - Flexible data model
    - RDF is a dynamic schema (so its built in!)

Express Yourself (Know your Language)

Practical lesson #5

- OWL
  - Its great for classification
    - Our business problem isn't about classification
  - First order logic
- Need solutions for non-classification logic
    - Time reasoning
    - Arithmetic logic
    - Counting items a collection



Text Service Not Working

Text Service Not Working

OWNER: Greg Sloan LAST MODIFIED: 12/10/2010 MODIFIED BY: Greg Sloan VERSION: 1.0 COMPUTED: Automatically

DESCRIPTION: Action to take when text service is not working. [Edit]

1. If Service Is Text And Activated At Least 3 Days Ago And No Messages Sent

Then Test Text Service

Express Yourself (Know your Language)

Practical lesson #5

- The semantic stack lacks a capable language for describing enterprise business logic
  - OWL is great for classification
  - SPARQL is not programming language
  - There is no stored procedure language
    - Choose a database that has a stored procedure language
- You have to find a solution
    - SPIN?
    - Prolog?
    - Existing rule engine?

The key is expressivity of your business problem

Modeling, Your New Career

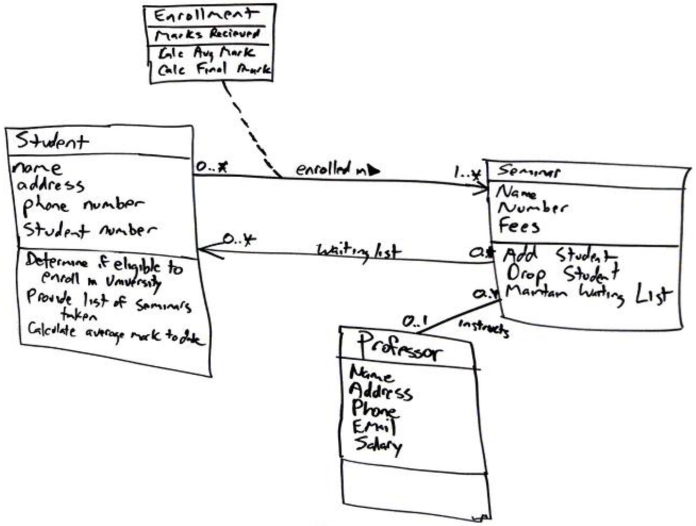
Practical lesson #6

Create simple, consistent models

Use patterns

Modeling, Your New Career

Practical lesson #6



- Don’t start over
  - Import your entities and fix them
    - Output as UML
    - Convert the UML to ontology's
      - In your UML model two classes might have an attribute with the same name
        - Top braid names it status 1, status 2
      - Want separate properties with different ranges and values
      - Change them in the Model
        - customer status customer account status

- In Enterprise applications you are modeling an object model
  - RDF basics cover everything you will likely need
    - Properties stand alone, can be used across classes.
      - Different range they are different things
      - Allows a property to be an A or B
    - Share properties across classes
      - Example -Name

Modeling, Your New Career

Practical lesson #6

- Strings
  - Don’t do operations on the string
    - Subject is string
      - Contains Type,Region,ID
    - Since Its overloaded processing is complex
      - Select every triple and filter

- Patterns are critical
  - History order date
  - List of values
    - Natural thing in RDF
  - Ordering collections
  - Finding the latest one
  - Keeping history of changes

Develop your patterns
Use them

Modeling, Your New Career

Practical lesson #6

- Namespaces
  - Create your namespace organization
  - Built in support allows for customizations preserving the core
    - Natural mechanism for in field customization

- Use the graph (if your DB supports it)
  - If you have a high number of queries
  - Specify the graph for each instance
  - History graph (graph per month)
    - Drop graph as a delete strategy
    - Contemporary

Modeling, Your New Career

Practical lesson #6

- Keep the ontology simple with low levels of indirection
- Good engineering for triples
  - Keep ontology simple
  - Add triples to create a natural relationship
  - Use types
    - If you have datatypes it Allows range queries
  - Enriched your data with annotations
    - In our system annotations drive the behavior of the system
      - Transient or persistent, Automatic or on-demand computation,…
  - Use a human readable label for your last part of the triple
  - For very important queries us custom indexes

It’s a Real Project

Practical lesson #7

Execute it like it’s a an enterprise project

It’s a Real Project

Practical lesson #7

- Far too many semantic projects begin and end in the lab
- Use all of the project discipline
  - Define success
  - Train the team
  - Develop the requirements
  - Plan the iterations
  - Test the functionality
  - Test the performance and scalability