Provider Registry System: HL7 v3 Interface

Andrew Cripps & Paolo Marcucci

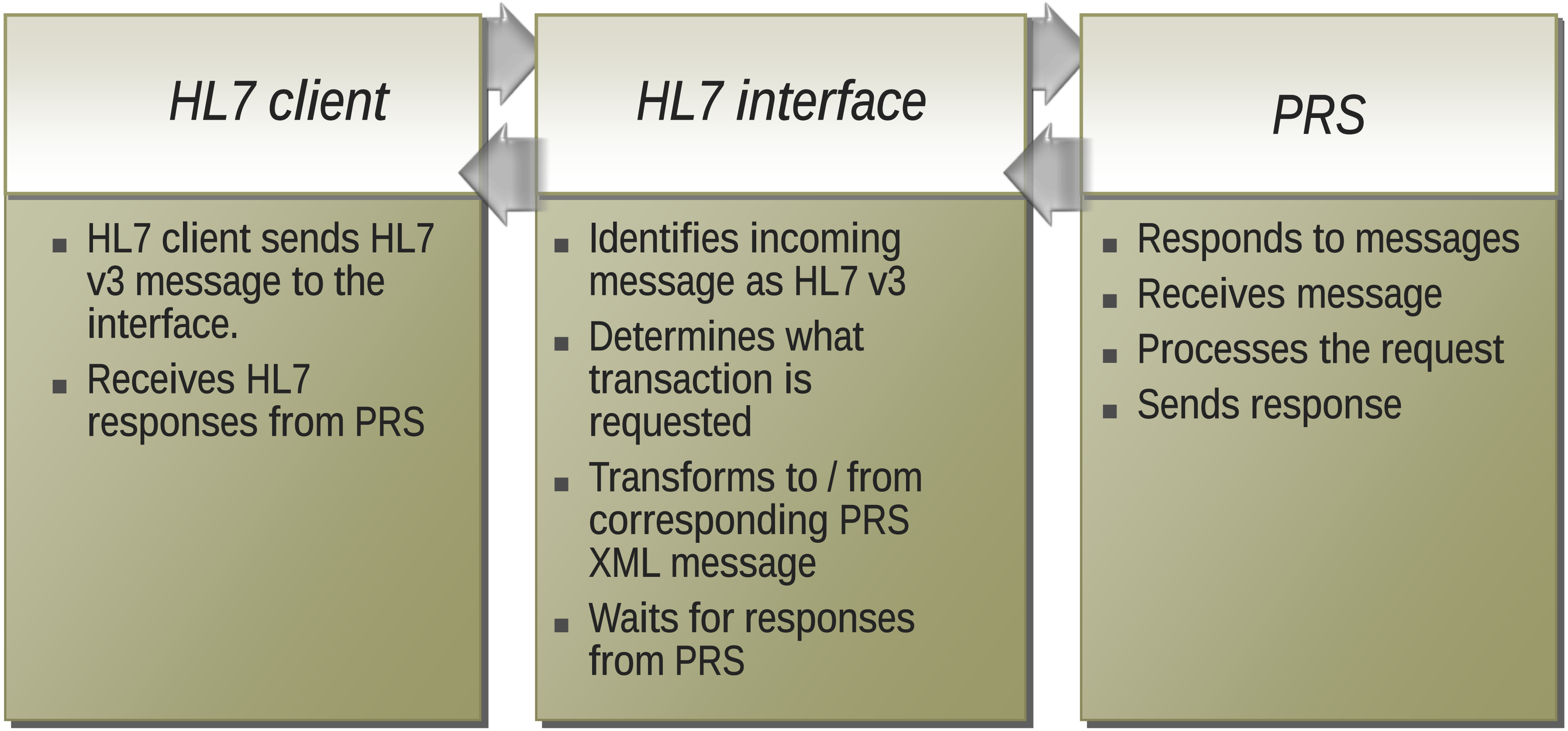
Sierra Systems

Sierra Systems

Introduction - Who

- Andrew Cripps
- Paolo Marcucci
- WHIC Provider Registry

Technical Design – Overview



HL7 client

- HL7 client sends HL7 v3 message to the interface.
- Receives HL7 responses from PRS

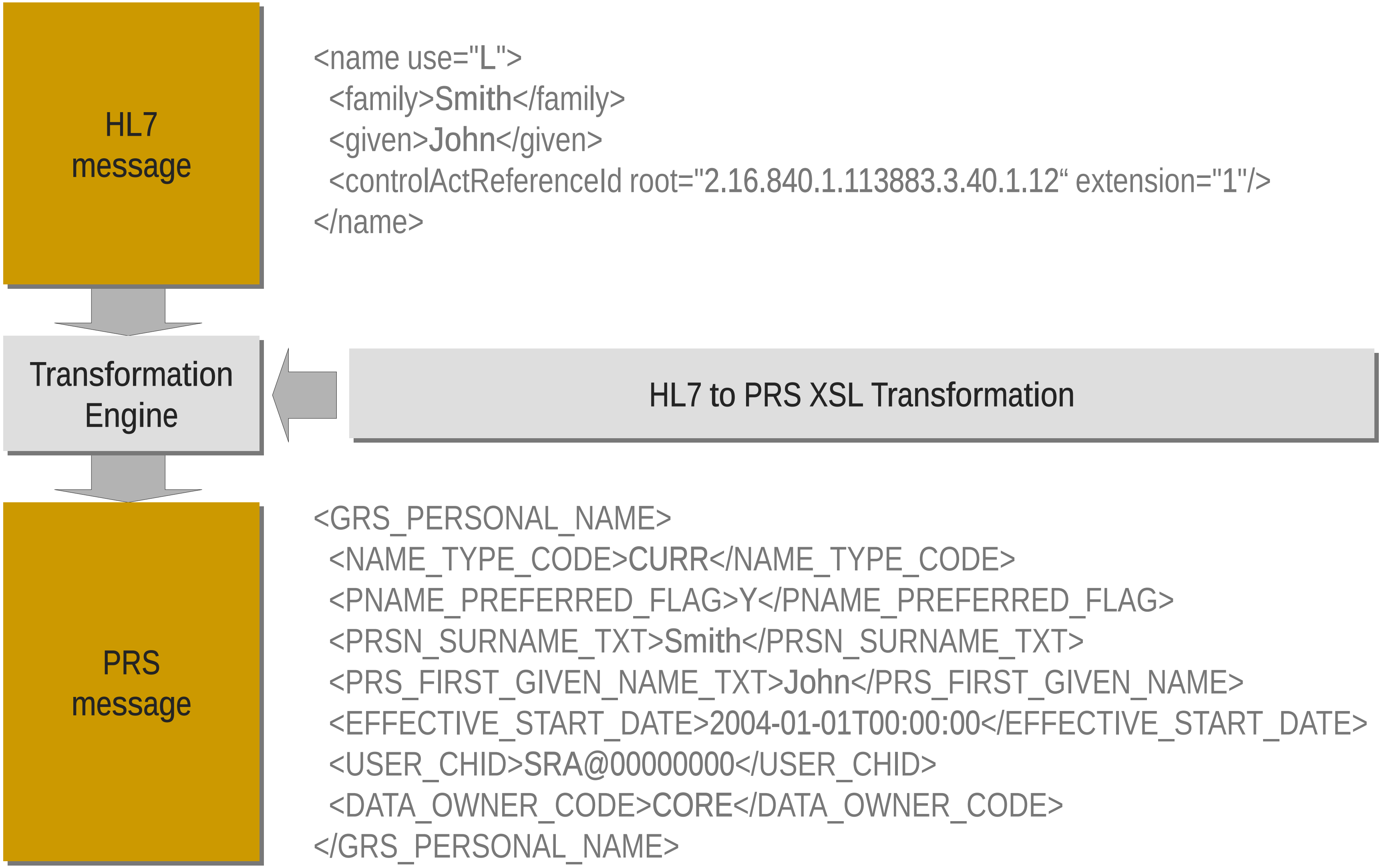
HL7 interface

- Identifies incoming message as HL7 v3
- Determines what transaction is requested
- Transforms to / from corresponding PRS XML message
- Waits for responses from PRS

PRS

- Responds to messages
- Receives message
- Processes the request
- Sends response

Extensible Style Sheet (XSL) Transformations (Recap)



HL7 message

Transformation Engine

HL7 to PRS XSL Transformation

PRS message

```
<name use="L">
  <family>Smith</family>
  <given>John</given>
  <controlActReferenceId root="2.16.840.1.113883.3.40.1.12“ extension="1"/>
</name>
```

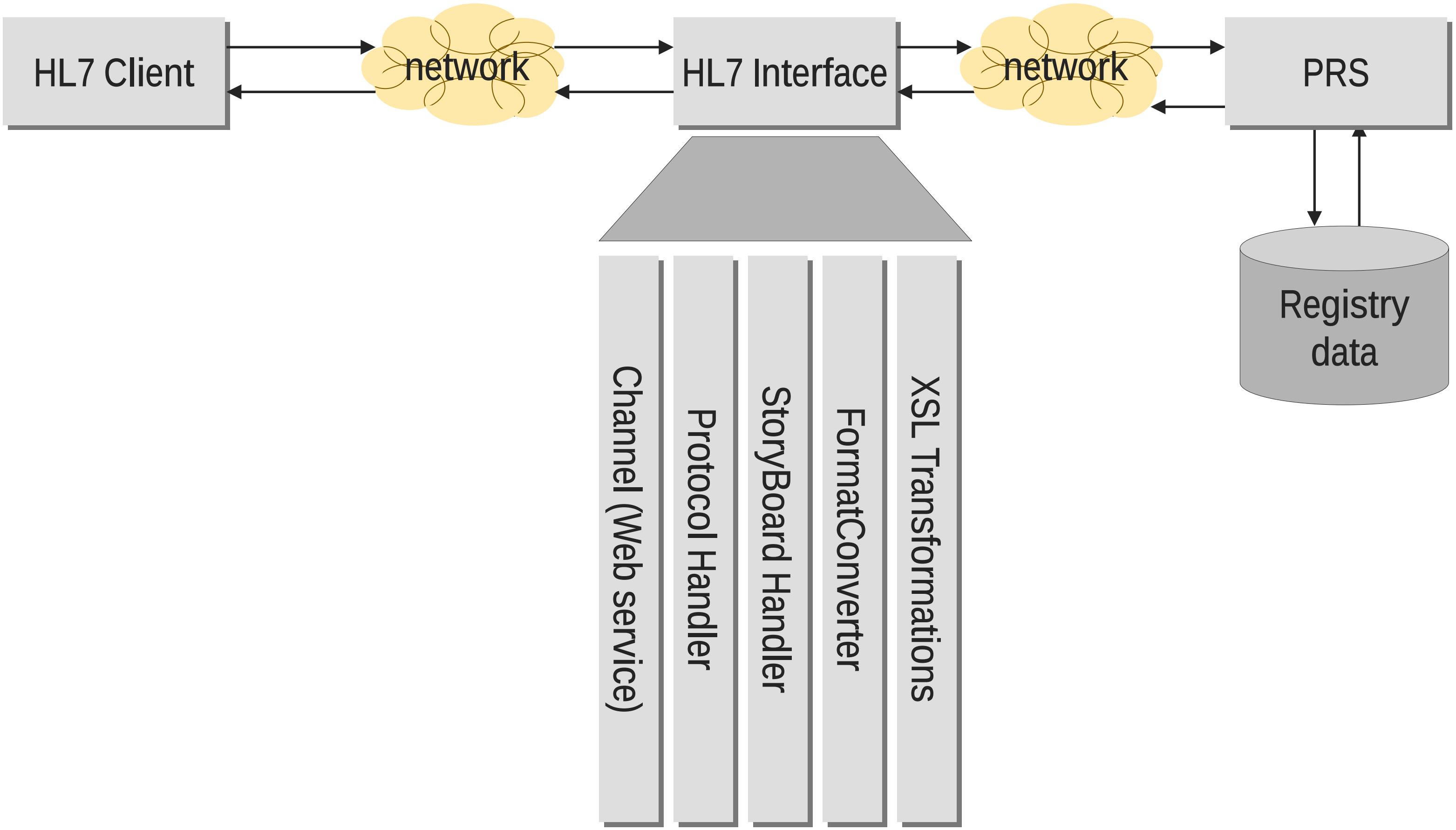
```
<GRS_PERSONAL_NAME>
  <NAME_TYPE_CODE>CURR</NAME_TYPE_CODE>
  <PNAME_PREFERRED_FLAG>Y</PNAME_PREFERRED_FLAG>
  <PRSN_SURNAME_TXT>Smith</PRSN_SURNAME_TXT>
  <PRS_FIRST_GIVEN_NAME_TXT>John</PRS_FIRST_GIVEN_NAME>
  <EFFECTIVE_START_DATE>2004-01-01T00:00:00</EFFECTIVE_START_DATE>
  <USER_CHID>SRA@00000000</USER_CHID>
  <DATA_OWNER_CODE>CORE</DATA_OWNER_CODE>
</GRS_PERSONAL_NAME>
```

Technical Design – Rationale

- HL7 concepts must be transformed to PRS concepts somewhere!
  - Either in the application between one data model and another;
  - Or at the message level

- Why did we choose transformations?
  - Separation from PRS
  - Ease of maintenance
  - No changes to the business or data layer

Technical Design – Detail



Technical Design

- External interface – receives HL7 messages, transforms, forwards
- PRS blackbox – responds to messages only
- Interface *transforms* between HL7 and PRS XML
- Can validate messages against the HL7 schemas

Iterative Process for Development

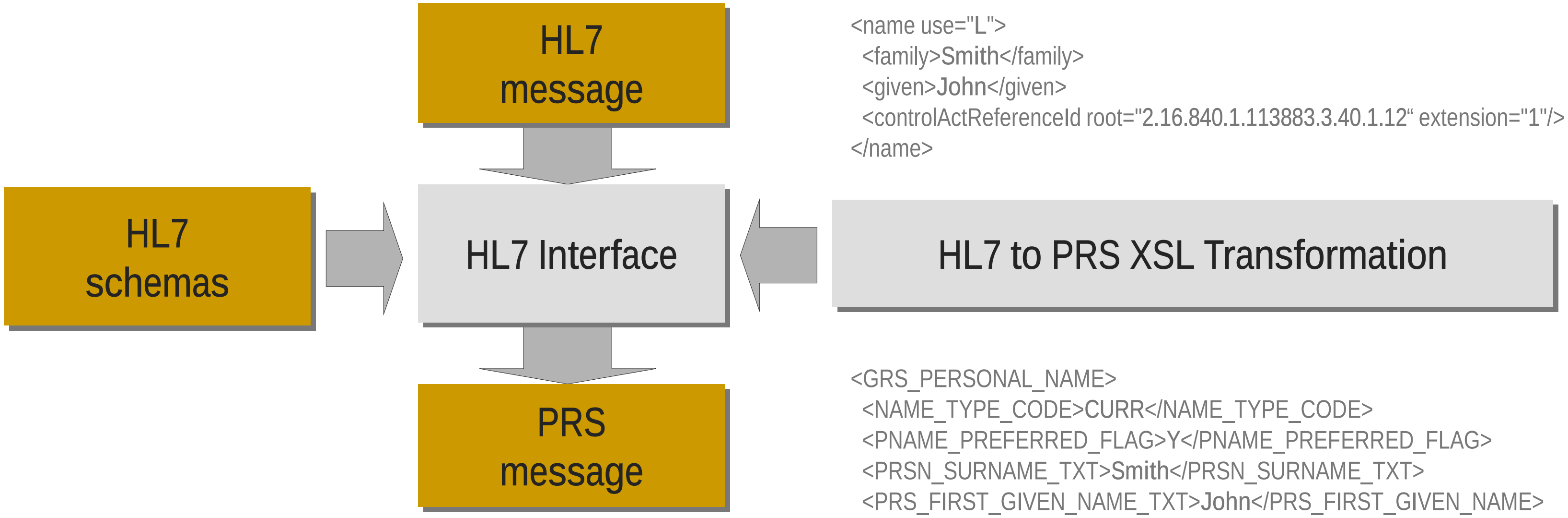
- Problem: The schema continued to be refined during development
  - This was an R & D effort
  - We are early adopters
- Critical factors for success:
  - Prototype project run to confirm feasibility
  - Identify changes required in the Provider Registry
  - Flexibility from all parties essential
  - Iterative development and testing essential
  - Create a way to refine message transformation easily

Design Approach

- Create mappings between PRS messages and XML messages
- Approach
  - Either use existing tools,
  - Or develop a code generator to produce transformations
- Code Generator chosen because
  - More control over what is produced
  - Current tools do not offer strong support for complex mappings

Design Approach – A note on Validation

- Use the interface in validation mode when testing
- In Production, validation would usually be turned off

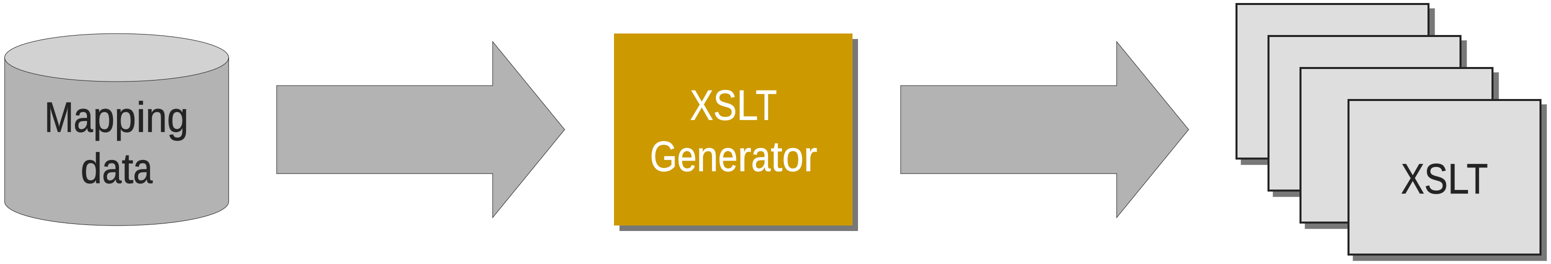


```
<name use="L">
  <family>Smith</family>
  <given>John</given>
  <controlActReferenceId root="2.16.840.1.113883.3.40.1.12“ extension="1"/>
</name>
```

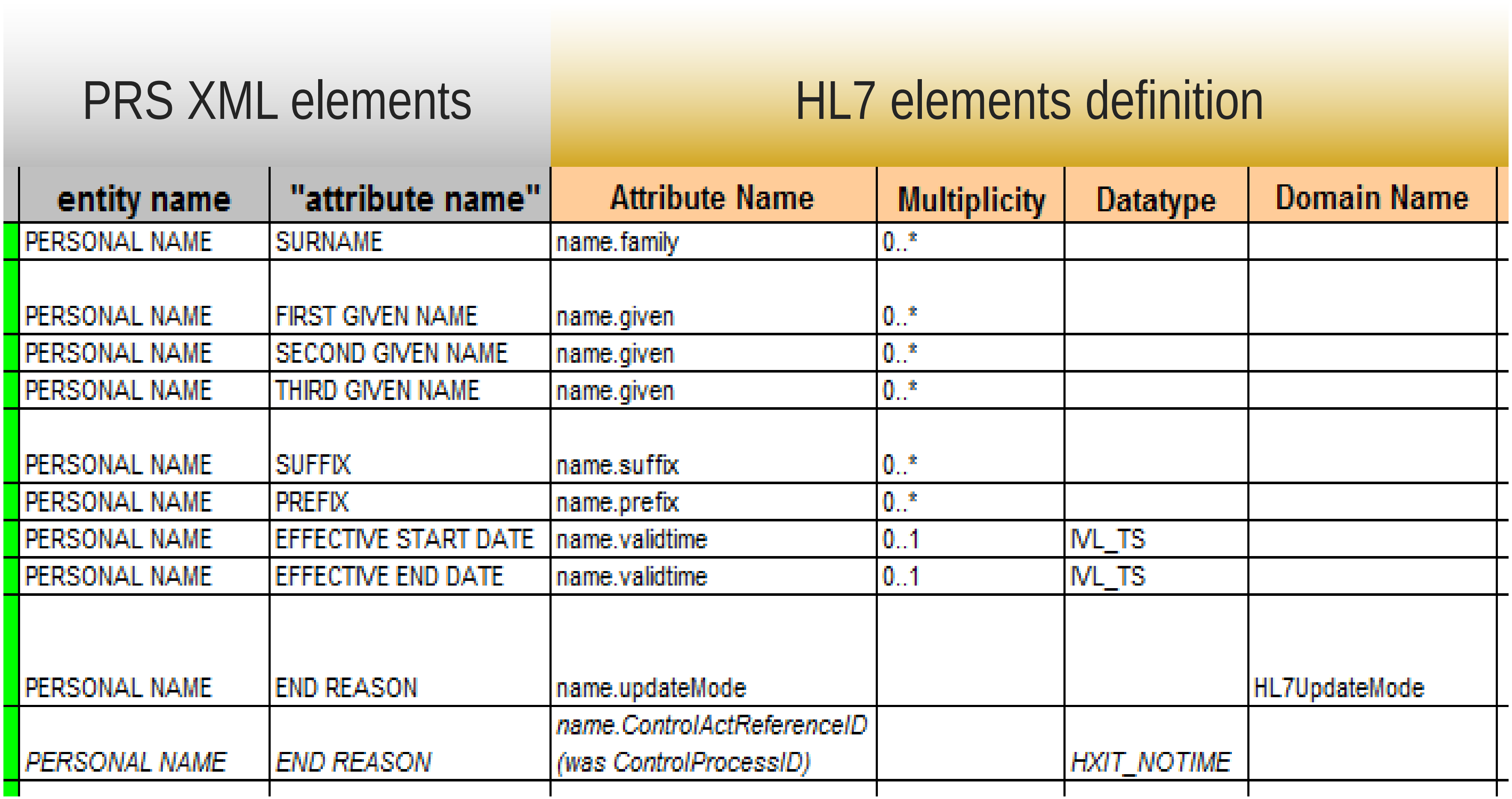
```
<GRS_PERSONAL_NAME>
  <NAME_TYPE_CODE>CURR</NAME_TYPE_CODE>
  <PNAME_PREFERRED_FLAG>Y</PNAME_PREFERRED_FLAG>
  <PRSN_SURNAME_TXT>Smith</PRSN_SURNAME_TXT>
  <PRS_FIRST_GIVEN_NAME_TXT>John</PRS_FIRST_GIVEN_NAME>
  <EFFECTIVE_START_DATE>2004-01-01T00:00:00</EFFECTIVE_START_DATE>
  <USER_CHID>SRA@00000000</USER_CHID>
  <DATA_OWNER_CODE>CORE</DATA_OWNER_CODE>
</GRS_PERSONAL_NAME>
```

XSLT Generator

- Objective: Produce transformations between HL7 and PRS messages
- Mappings housed in an Access database
- Output of the generator
  - A set of XSL transformations corresponding to HL7 requests and responses

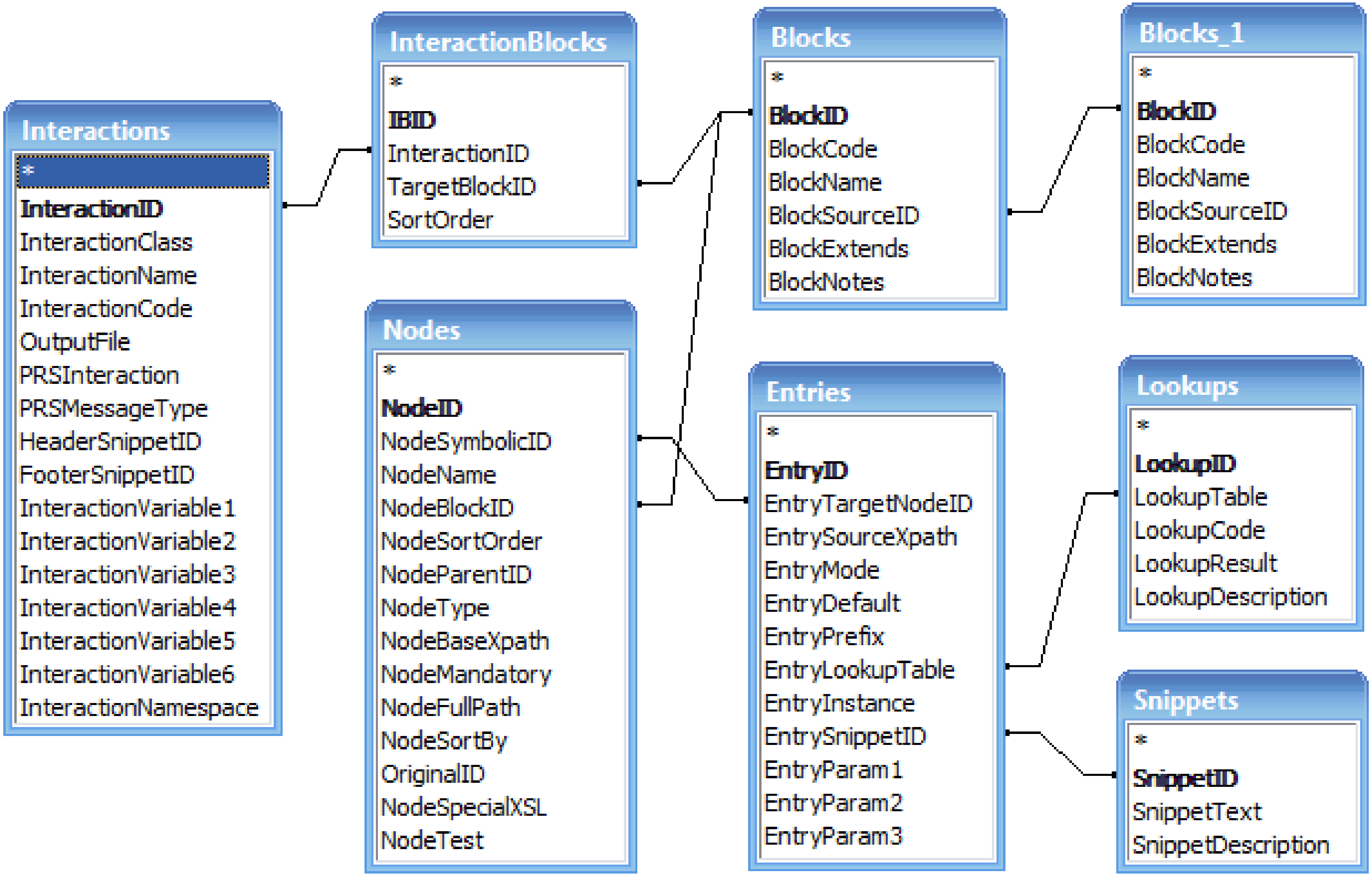


Mapping spreadsheet



PRS XML elements		HL7 elements definition			
entity name	**"attribute name"**	**Attribute Name**	**Multiplicity**	**Datatype**	**Domain Name**
PERSONAL NAME	SURNAME	name.family	0..*		
PERSONAL NAME	FIRST GIVEN NAME	name.given	0..*		
PERSONAL NAME	SECOND GIVEN NAME	name.given	0..*		
PERSONAL NAME	THIRD GIVEN NAME	name.given	0..*		
PERSONAL NAME	SUFFIX	name.suffix	0..*		
PERSONAL NAME	PREFIX	name.prefix	0..*		
PERSONAL NAME	EFFECTIVE START DATE	name.validtime	0..1	IVL_TS	
PERSONAL NAME	EFFECTIVE END DATE	name.validtime	0..1	IVL_TS	
PERSONAL NAME	END REASON	name.updateMode			HL7UpdateMode
PERSONAL NAME	*END REASON*	*name.ControlActReferenceID (was ControlProcessID)*		*HXIT_NOTIME*	

Elements repository



Interactions
*
InteractionID
InteractionClass
InteractionName
InteractionCode
OutputFile
PRSInteraction
PRSMessageType
HeaderSnippetID
FooterSnippetID
InteractionVariable1
InteractionVariable2
InteractionVariable3
InteractionVariable4
InteractionVariable5
InteractionVariable6
InteractionNamespace

InteractionBlocks
*
IBID
InteractionID
TargetBlockID
SortOrder

Nodes
*
NodeID
NodeSymbolicID
NodeName
NodeBlockID
NodeSortOrder
NodeParentID
NodeType
NodeBaseXpath
NodeMandatory
NodeFullPath
NodeSortBy
OriginalID
NodeSpecialXSL
NodeTest

Blocks
*
BlockID
BlockCode
BlockName
BlockSourceID
BlockExtends
BlockNotes

Entries
*
EntryID
EntryTargetNodeID
EntrySourceXpath
EntryMode
EntryDefault
EntryPrefix
EntryLookupTable
EntryInstance
EntrySnippetID
EntryParam1
EntryParam2
EntryParam3

Blocks_1
*
BlockID
BlockCode
BlockName
BlockSourceID
BlockExtends
BlockNotes

Lookups
*
LookupID
LookupTable
LookupCode
LookupResult
LookupDescription

Snippets
*
SnippetID
SnippetText
SnippetDescription

Generated XSLT fragment

```
<!-- Loop structure [GRS_PERSONAL_NAME] n:555 b:43 -->
<xsl:for-each
      select="n1:controlActProcess/n1:subject/n1:registrationEvent/n1:subject1/*/n1:healthCarePrincipalPerson/n1:name
      ">
 <!-- Element (inside a loop) [GRS_PERSONAL_NAME] n:555 b:43 -->
 <GRS_PERSONAL_NAME>
  <!-- Element [NAME_TYPE_CODE] n:556 b:43 e:465 -->
  <NAME_TYPE_CODE>
   <!-- Snippet (s:16 - Generic lookup) [NAME_TYPE_CODE] n:556 b:43 e:465 -->
   <xsl:variable name="var">
    <xsl:value-of select="@use"/>
   </xsl:variable>
   <xsl:choose>
    <xsl:when test="$var = 'C'">CRED</xsl:when>
    <xsl:when test="$var = 'L'">CURR</xsl:when>
    <xsl:otherwise>--</xsl:otherwise>
   </xsl:choose>
  </NAME_TYPE_CODE>
```

HL7 input elements

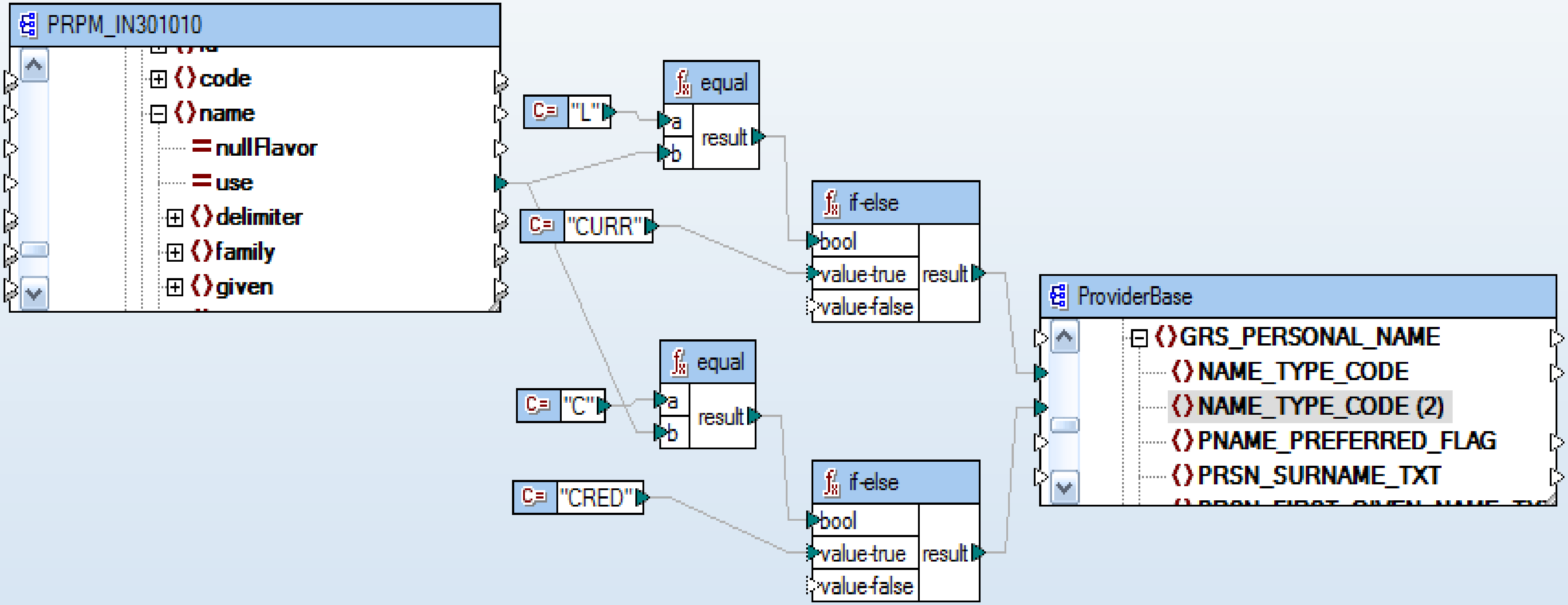
XSLT processing instructions

PRS XML output elements

Sierra Systems

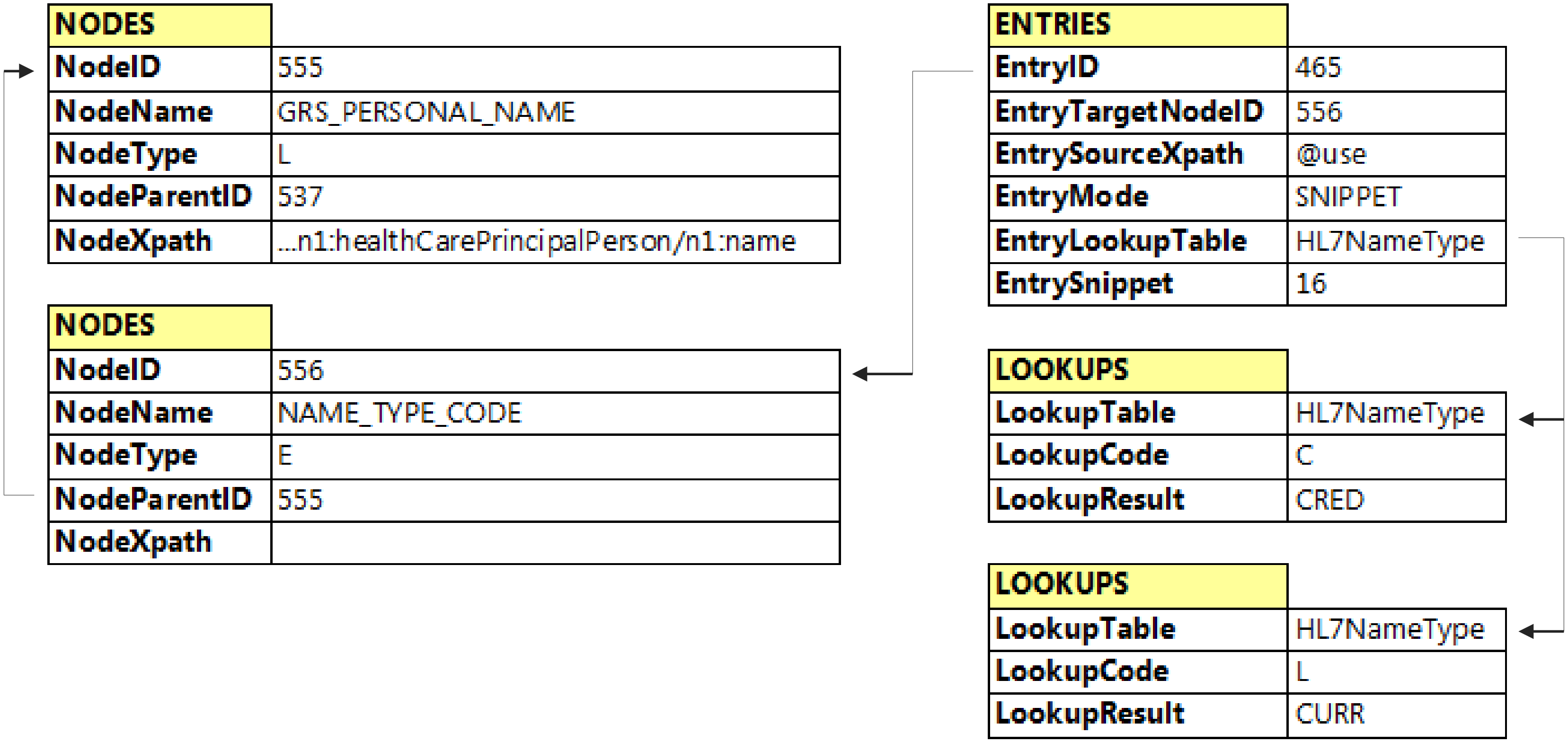
Mapping tool example

- Altova MapForce 2005



Elements repository extract

- Five rows in three tables to define this transformation fragment for all HL7 messages



NODES	
NodeID	555
NodeName	GRS_PERSONAL_NAME
NodeType	L
NodeParentID	537
NodeXpath	...n1:healthCarePrincipalPerson/n1:name

NODES	
NodeID	556
NodeName	NAME_TYPE_CODE
NodeType	E
NodeParentID	555
NodeXpath	

ENTRIES	
EntryID	465
EntryTargetNodeID	556
EntrySourceXpath	@use
EntryMode	SNIPPET
EntryLookupTable	HL7NameType
EntrySnippet	16

LOOKUPS	
LookupTable	HL7NameType
LookupCode	C
LookupResult	CRED

LOOKUPS	
LookupTable	HL7NameType
LookupCode	L
LookupResult	CURR

Summary

- Design around constant change
  - Schemas will vary
  - Waterfall approach is not feasible
  - Establish controlled team communications

- Our solution can be reused
  - Adding an external interface layer is a valid approach
  - The XSLT Generator may be used for other message definitions