OPENEHR IN SWEDEN AND BEYOND

Thomas Beale, Lund 3 Nov 2010

What *open*EHR offers - the big picture

*open*EHR and process

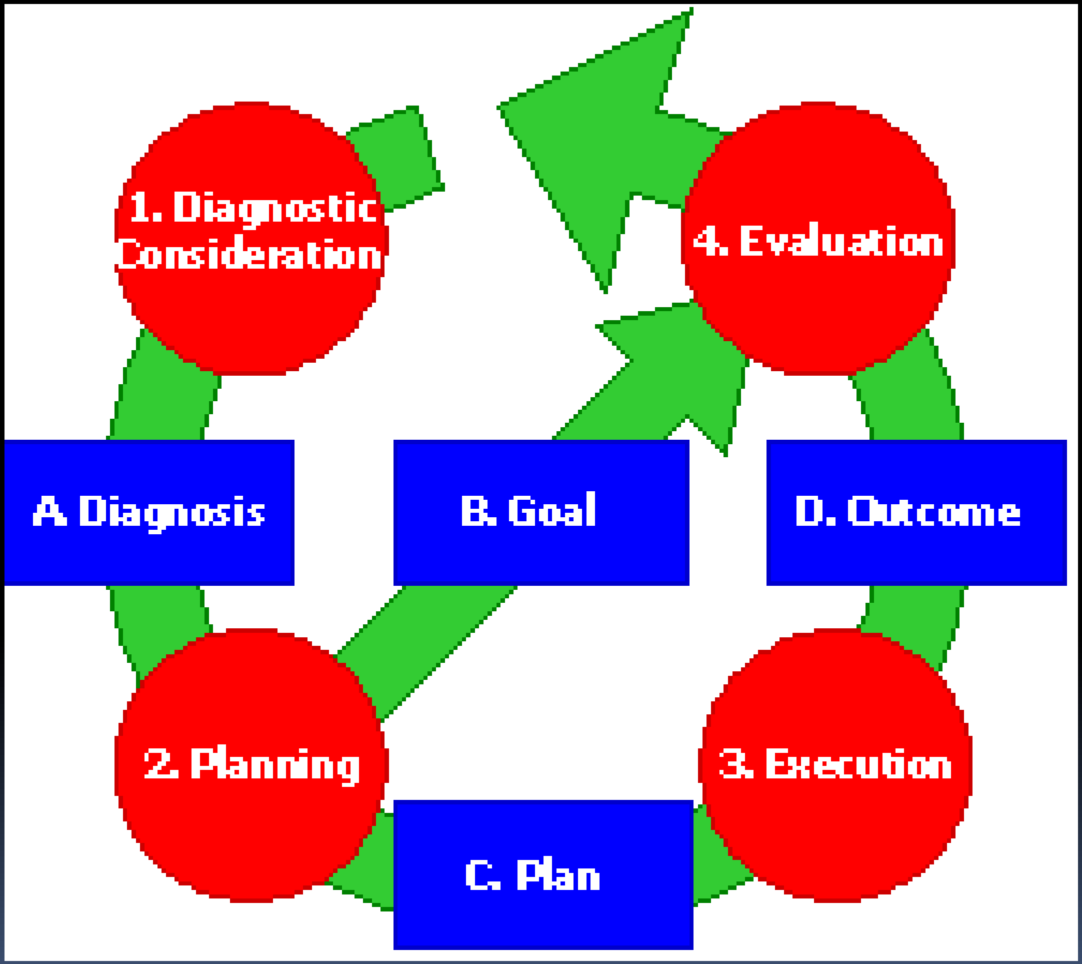
Our basis

- Clinical healthcare is a 1) rational scientific 2) problem-solving 3) process used to generate decisions
- Decisions require evidence and evaluation of evidence
- But real healthcare is messy
  - GPs may prescribe without diagnosing
  - Patients and nurses administer with no order
  - Many exceptions, e.g. reactions to drugs
  - Experienced doctors correctly diagnose without ‘following the book’

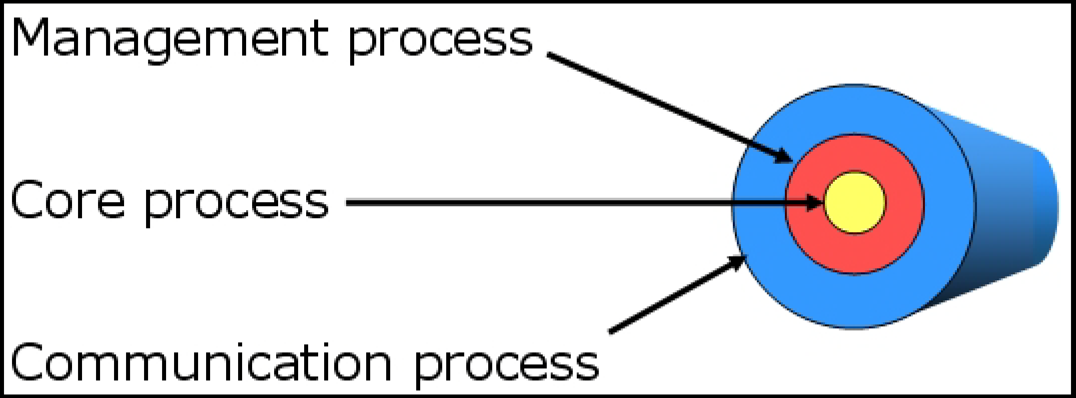
History of Solutions

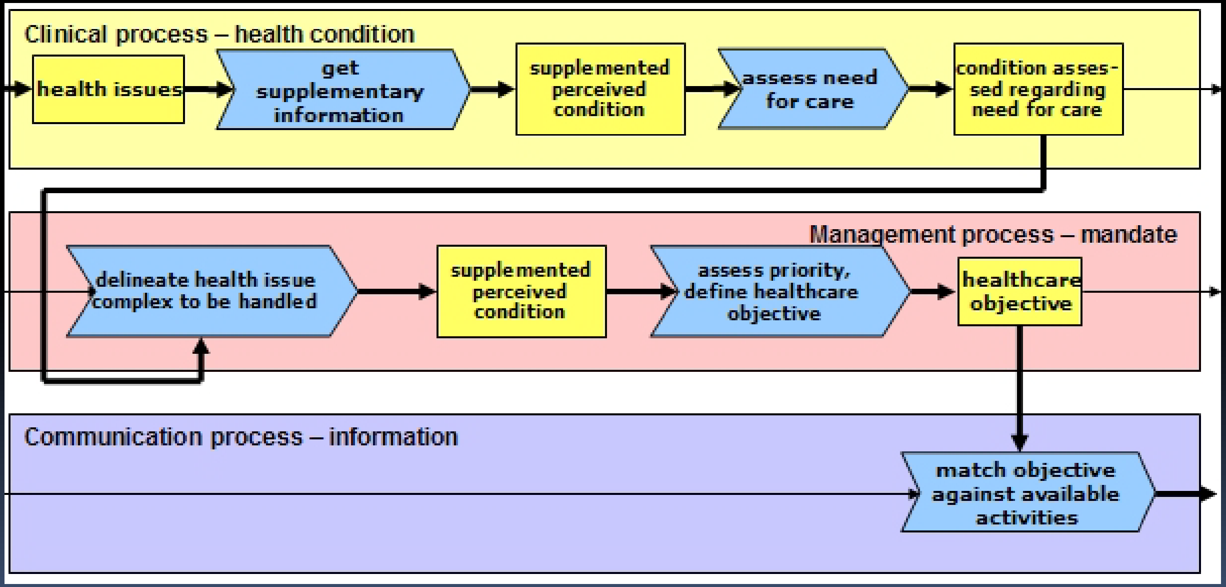
- Paper records – little internal organisation
- Weed’s POMR – SOAP organisation of information – hard to implement
- Elstein ‘hypothetico-deductive’ model of clinical reasoning – diagnosis-focussed
- Rector *et al* - PEN&PAD – how to record what we said, what was thought and what should be done about it

History of Solutions – Danish G-EPJ



History of Solutions - Samba





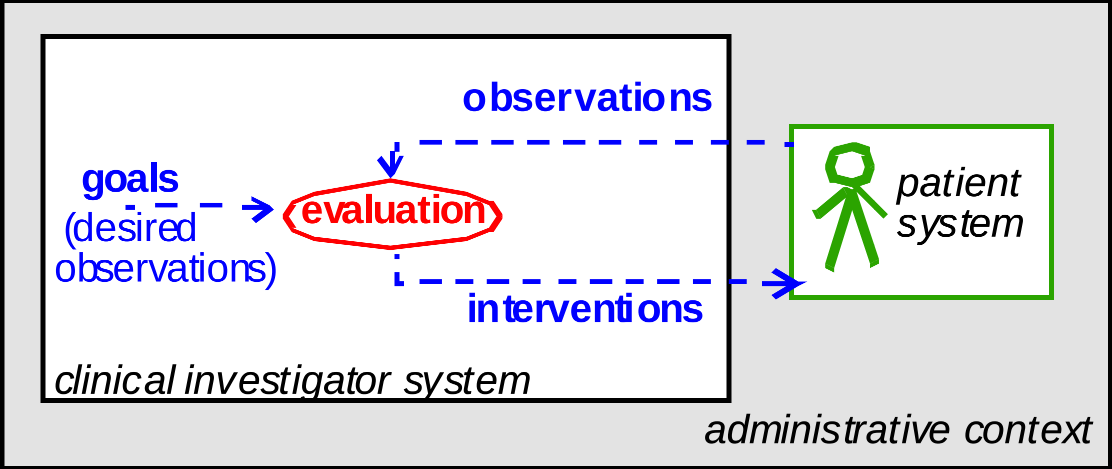
Clinical process – health condition
health issues
get supplementary information
supplemented perceived condition
assess need for care
condition assessed regarding need for care
Management process – mandate
delineate health issue complex to be handled
supplemented perceived condition
assess priority, define healthcare objective
healthcare objective
Communication process – information
match objective against available activities

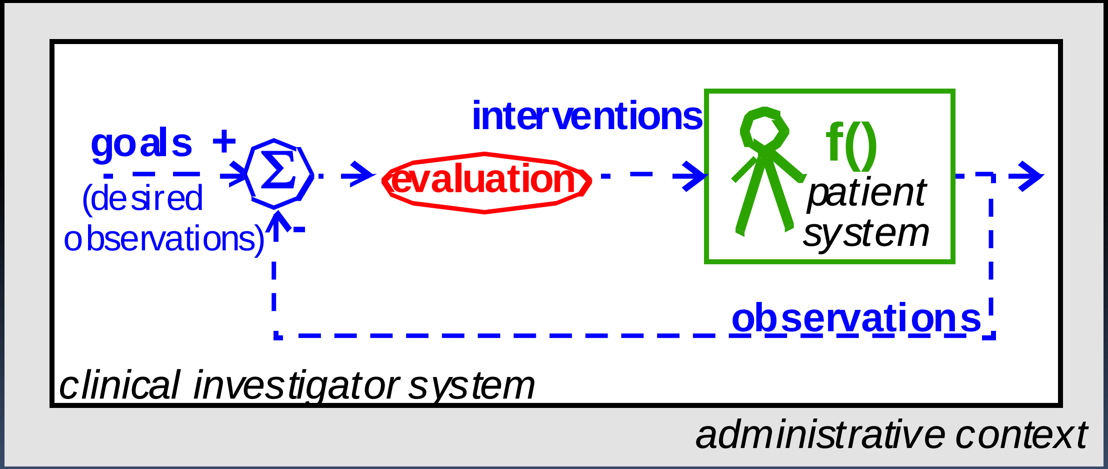
History of Solutions – Act-based

- Includes
  - RICHE
  - HL7v3 RIM
  - Many others
- Problems
  - everything is an act – good for tracking business process steps, but not natural to physicians
  - Hard to model typical clinical recordings

Our approach – 'Clinical Investigator'

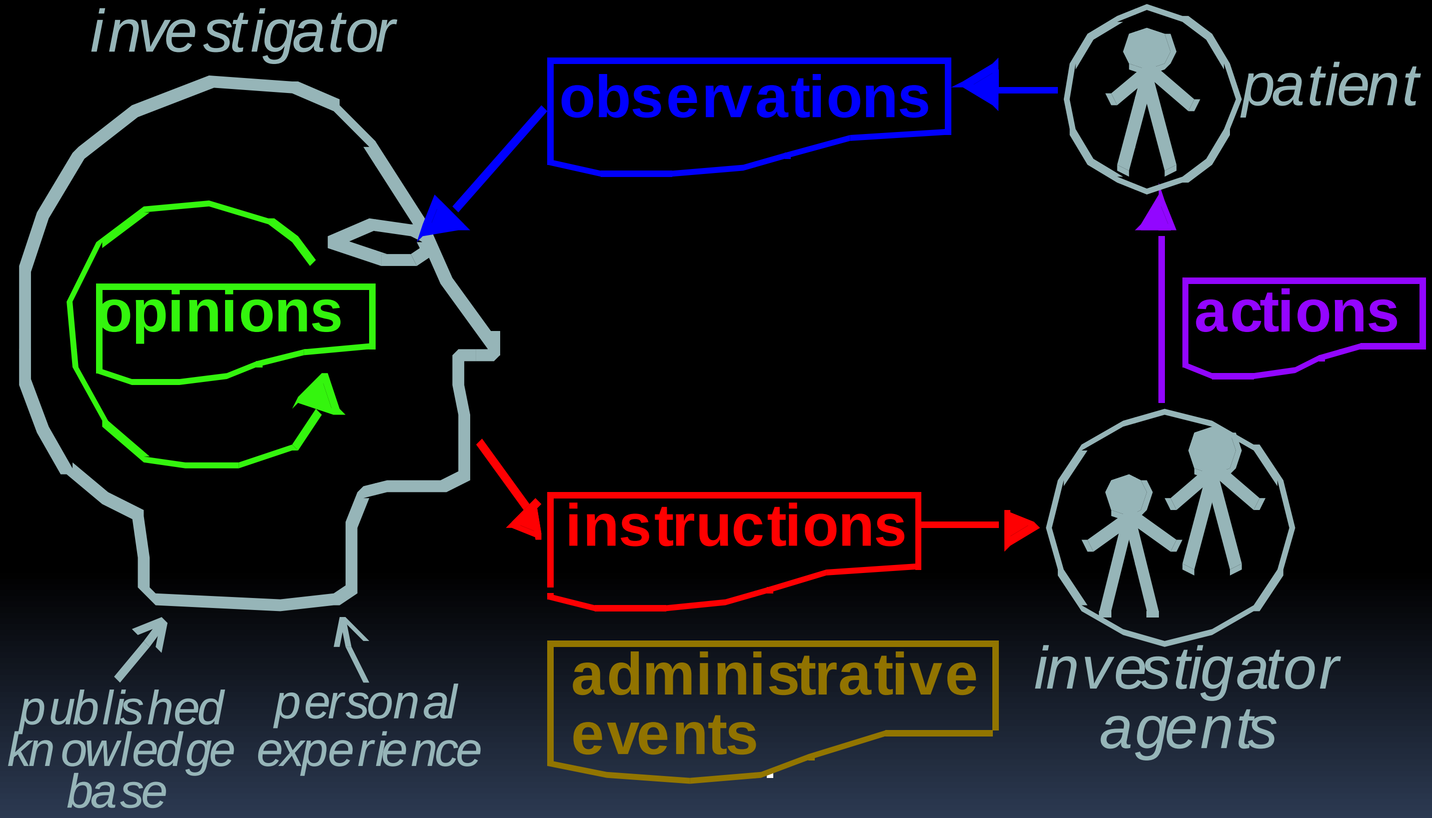
- Based on clinical process



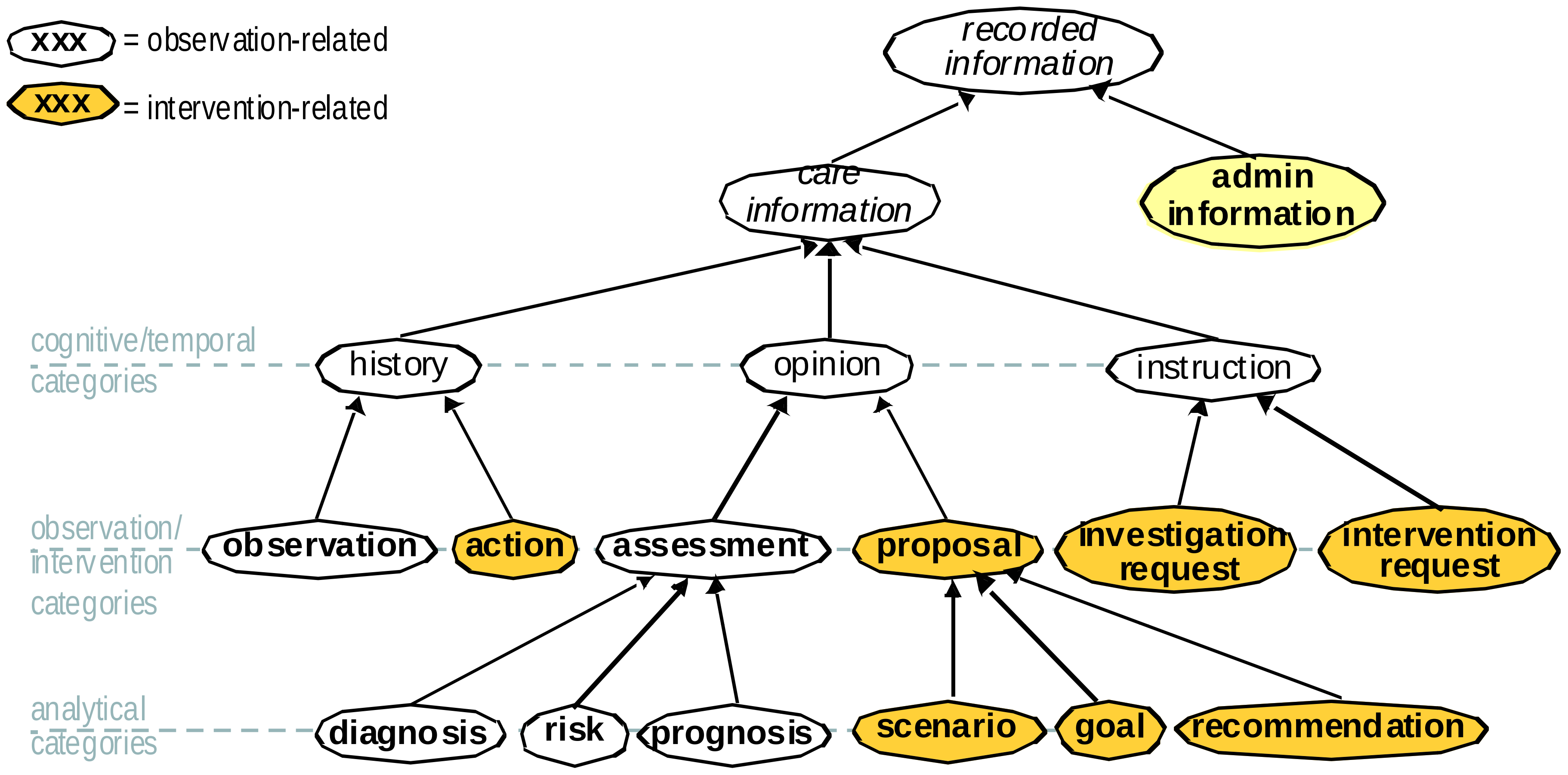


b) control system metaphor

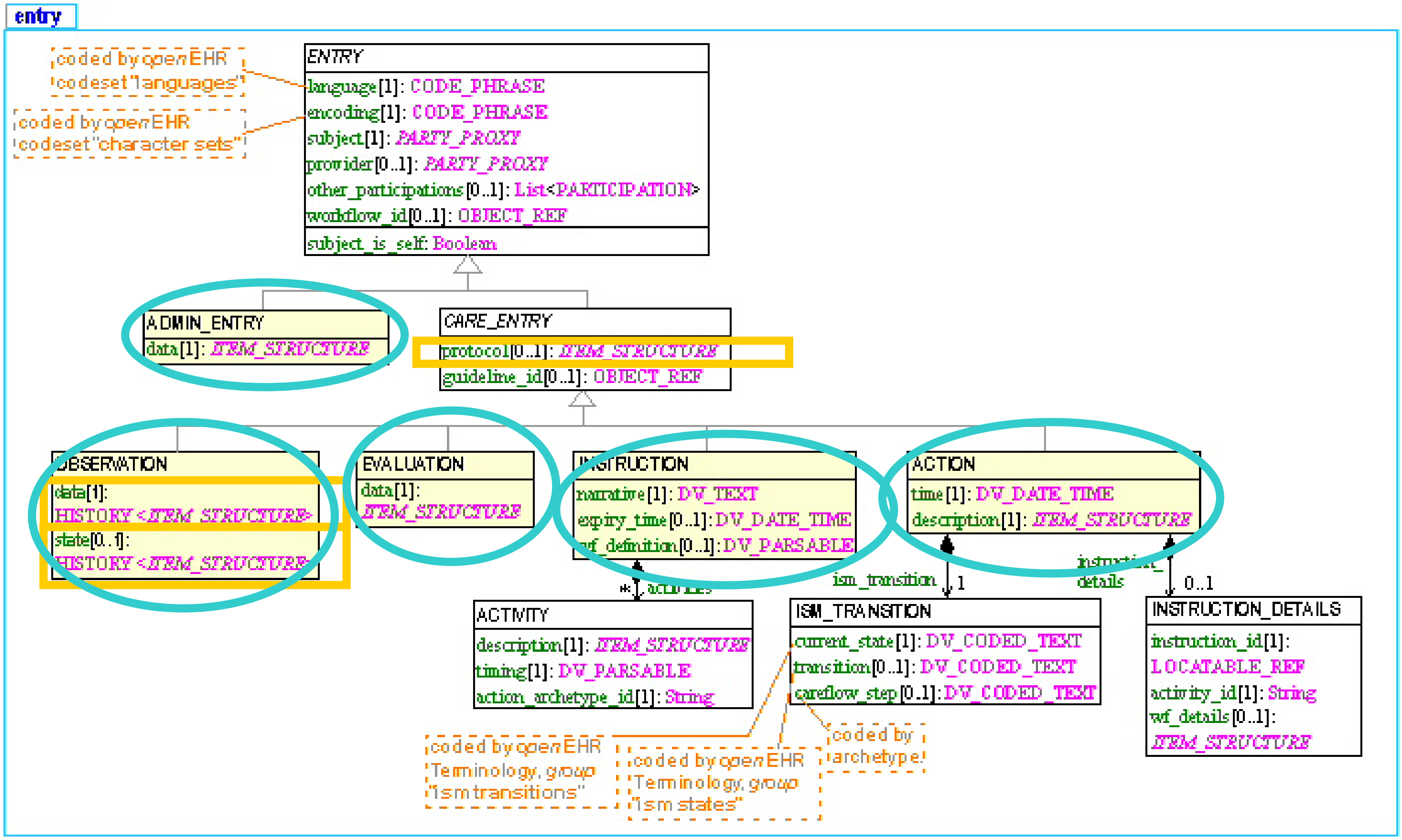
Leading to Types of Information



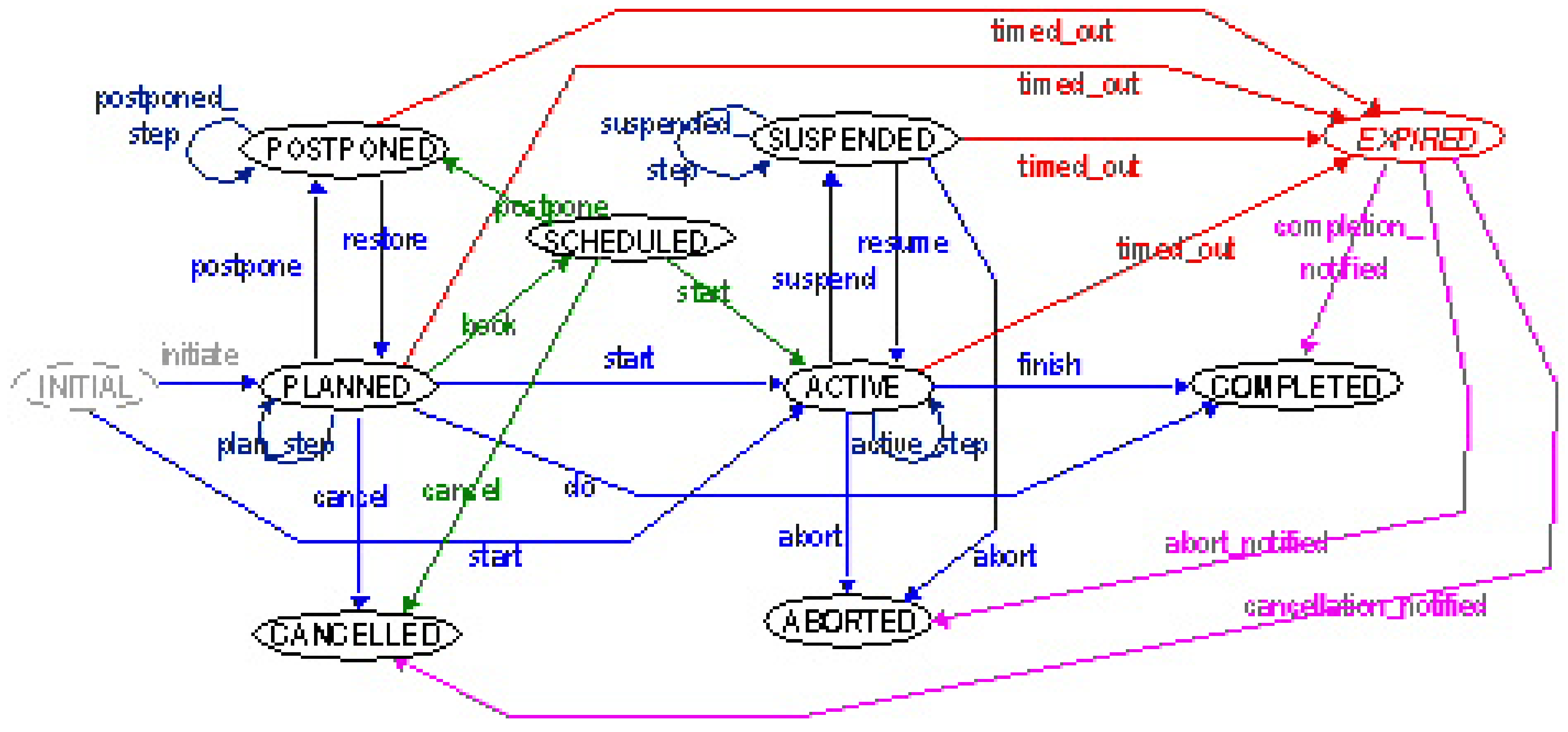
Leading to an Ontology



Leading to an Information Model
(Entry part shown here)


entry
coded by openEHR codeset "languages"
coded by openEHR codeset "character sets"
ENTRY
language[1]: CODE_PHRASE
encoding[1]: CODE_PHRASE
subject[1]: PARTY_PROXY
provider[0..1]: PARTY_PROXY
other_participations[0..1]: List<PARTICIPATION>
workflow_id[0..1]: OBJECT_REF
subject_is_self: Boolean
ADMIN_ENTRY
data[1]: ITEM_STRUCTURE
CARE_ENTRY
protocol[0..1]: ITEM_STRUCTURE
guideline_id[0..1]: OBJECT_REF
OBSERVATION
data[1]: HISTORY<ITEM_STRUCTURE>
state[0..1]: HISTORY<ITEM_STRUCTURE>
EVALUATION
data[1]: ITEM_STRUCTURE
INSTRUCTION
narrative[1]: DV_TEXT
expiry_time[0..1]: DV_DATE_TIME
wf_definition[0..1]: DV_PARSABLE
ACTION
time[1]: DV_DATE_TIME
description[1]: ITEM_STRUCTURE
* activities
ism_transition 1
instruction_details 0..1
ACTIVITY
description[1]: ITEM_STRUCTURE
timing[1]: DV_PARSABLE
action_archetype_id[1]: String
ISM_TRANSITION
current_state[1]: DV_CODED_TEXT
transition[0..1]: DV_CODED_TEXT
careflow_step[0..1]: DV_CODED_TEXT
INSTRUCTION_DETAILS
instruction_id[1]: LOCATABLE_REF
activity_id[1]: String
wf_details[0..1]: ITEM_STRUCTURE
coded by openEHR Terminology, group "ism transitions"
coded by openEHR Terminology, group "ism states"
coded by archetype!

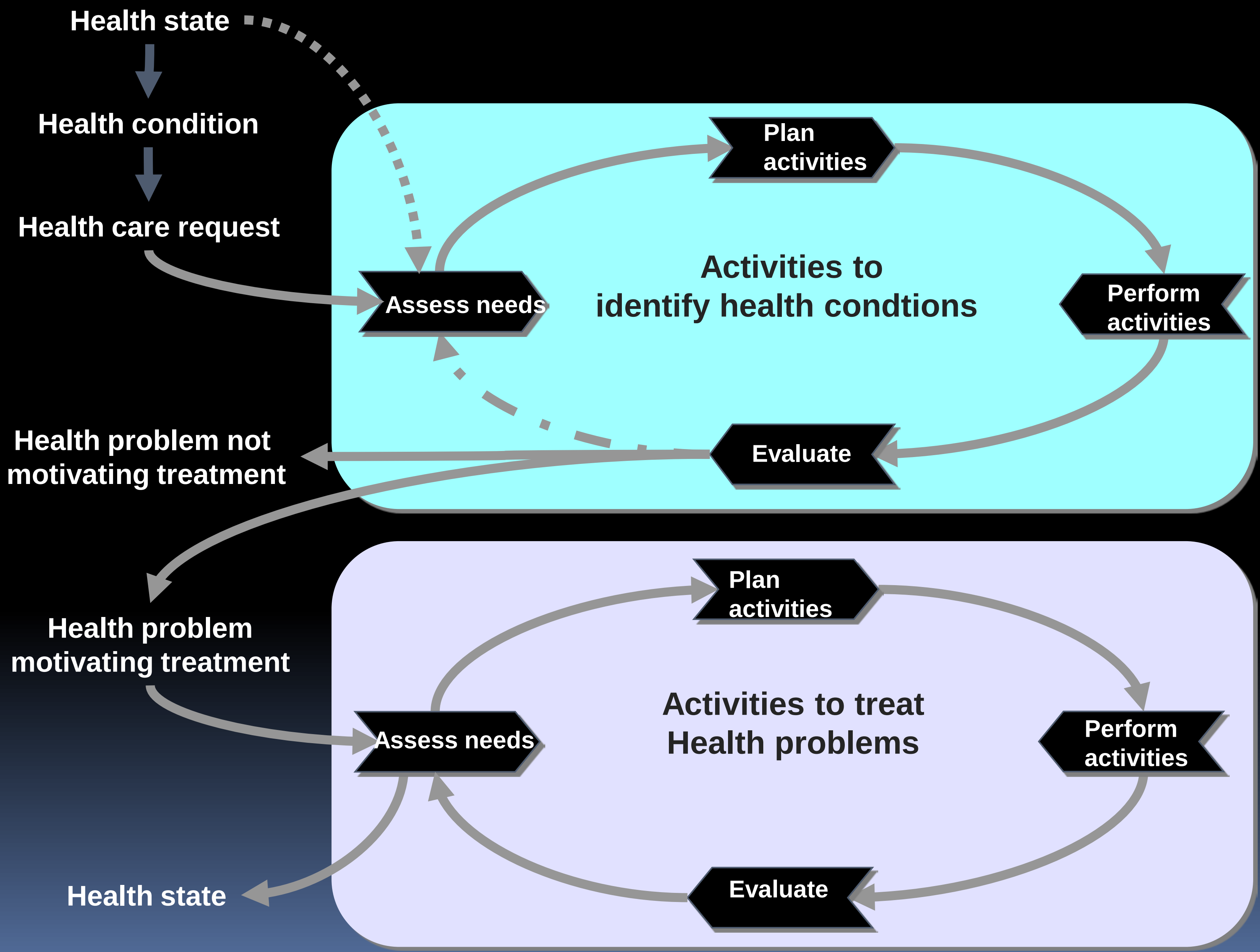
State machine – Instructions and Actions



Transition legend

blue: normal state changing event
dark blue: normal non-state changing event
green: scheduling-related transitions
red: time-out event
magenta: post time-out event

Swedish view

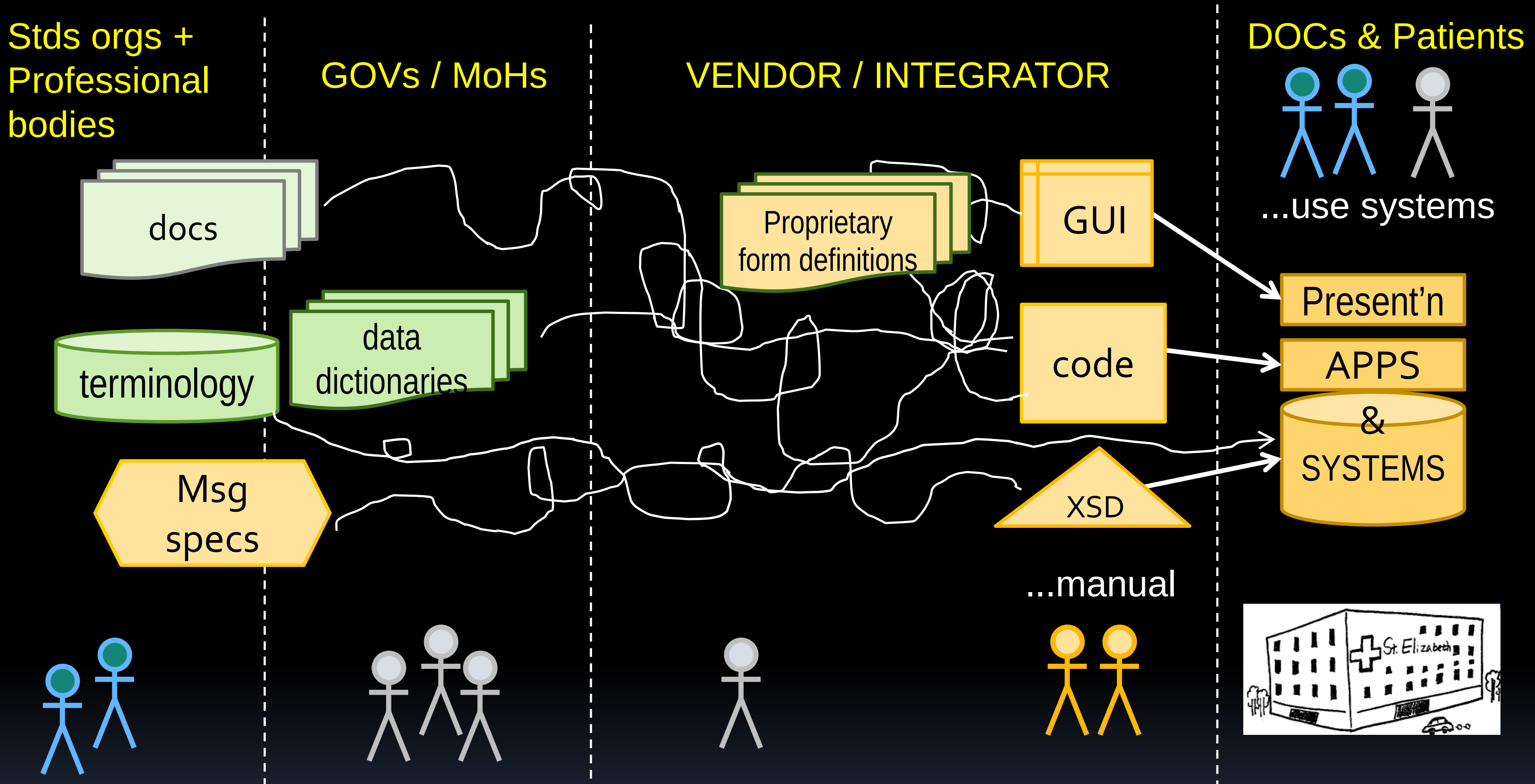


Conclusions

- Information types modelled
- LINKs, DV_EHR_URI type allow process-based referencing
- But controlling this with archetypes still not completely understood…
- What information structures do we want in the clinical data repository?
- SEE: http://www.openehr.org/publications/health_ict/MedInfo2007-BealeHeard.pdf

The ecosystem

Historical Industry Structure



Write docs, create message specs

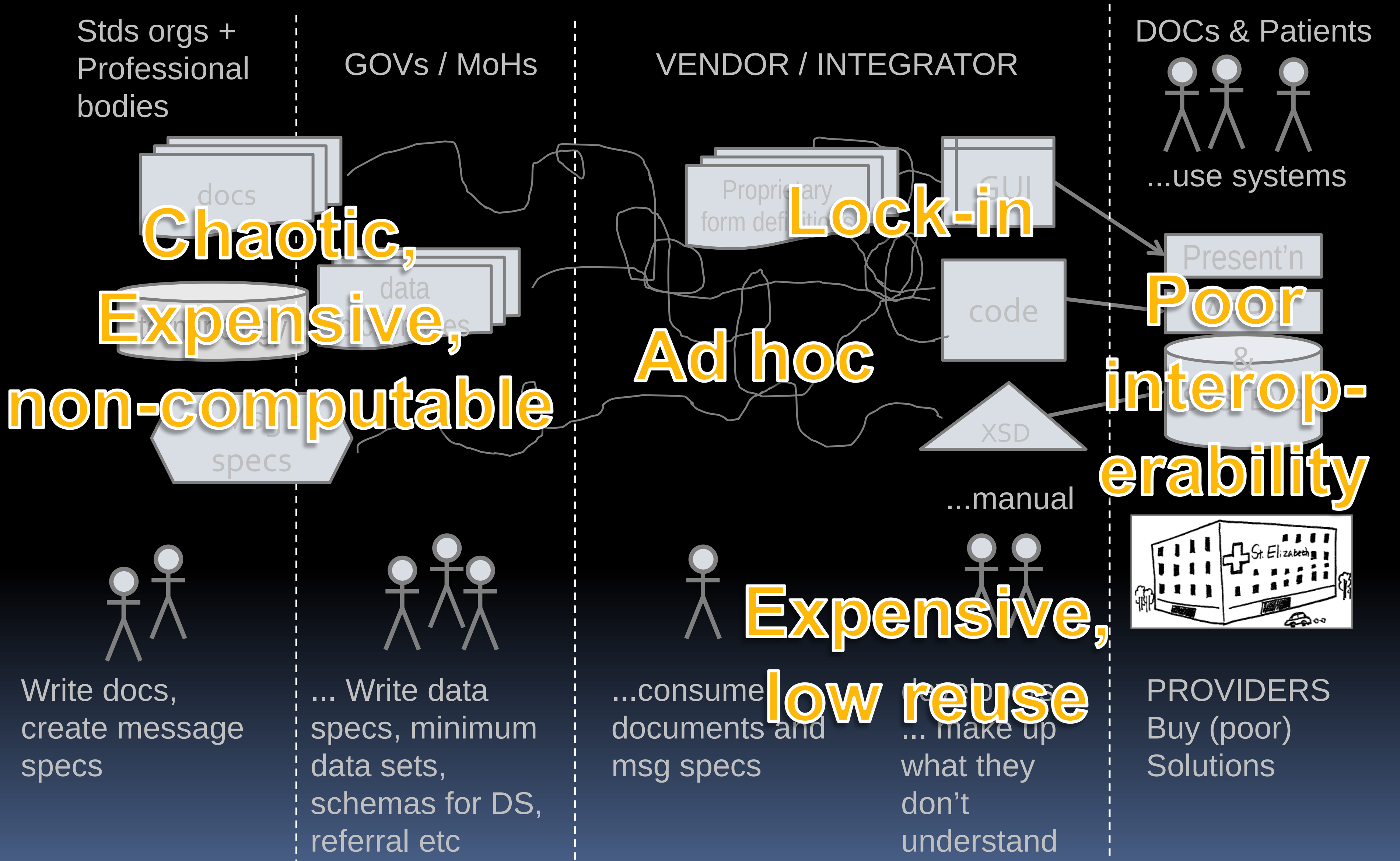
... Write data specs, minimum data sets, schemas for DS, referral etc

...consume documents and msg specs

developers ... make up what they don't understand

PROVIDERS Buy (poor) Solutions

Historical Industry Structure



Write docs, create message specs

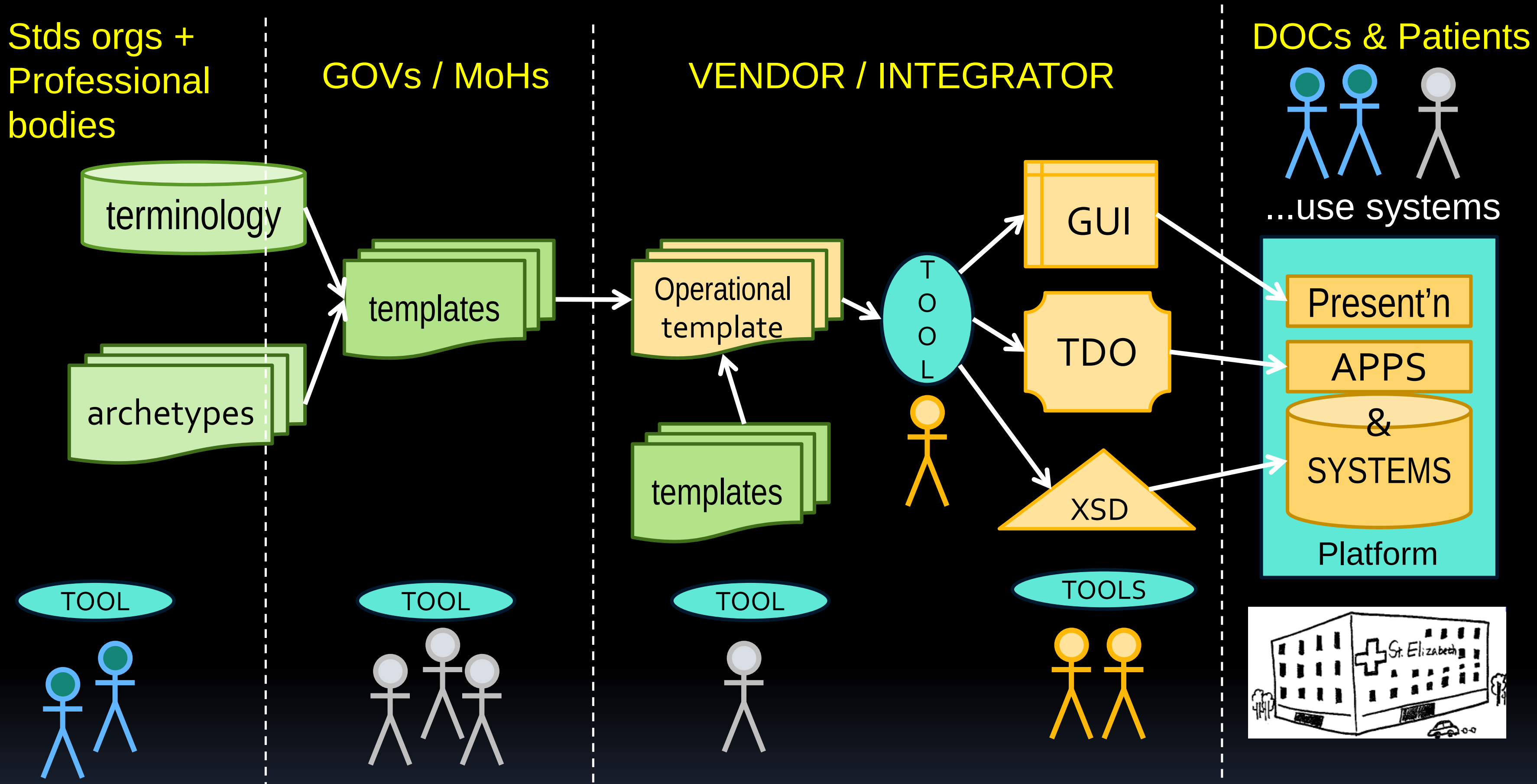
... Write data specs, minimum data sets, schemas for DS, referral etc

...consume documents and msg specs

... developers ... make up what they don't understand

PROVIDERS Buy (poor) Solutions

*open*EHR approach



build archetypes & terminology that define their Information – e.g. via IHTSDO

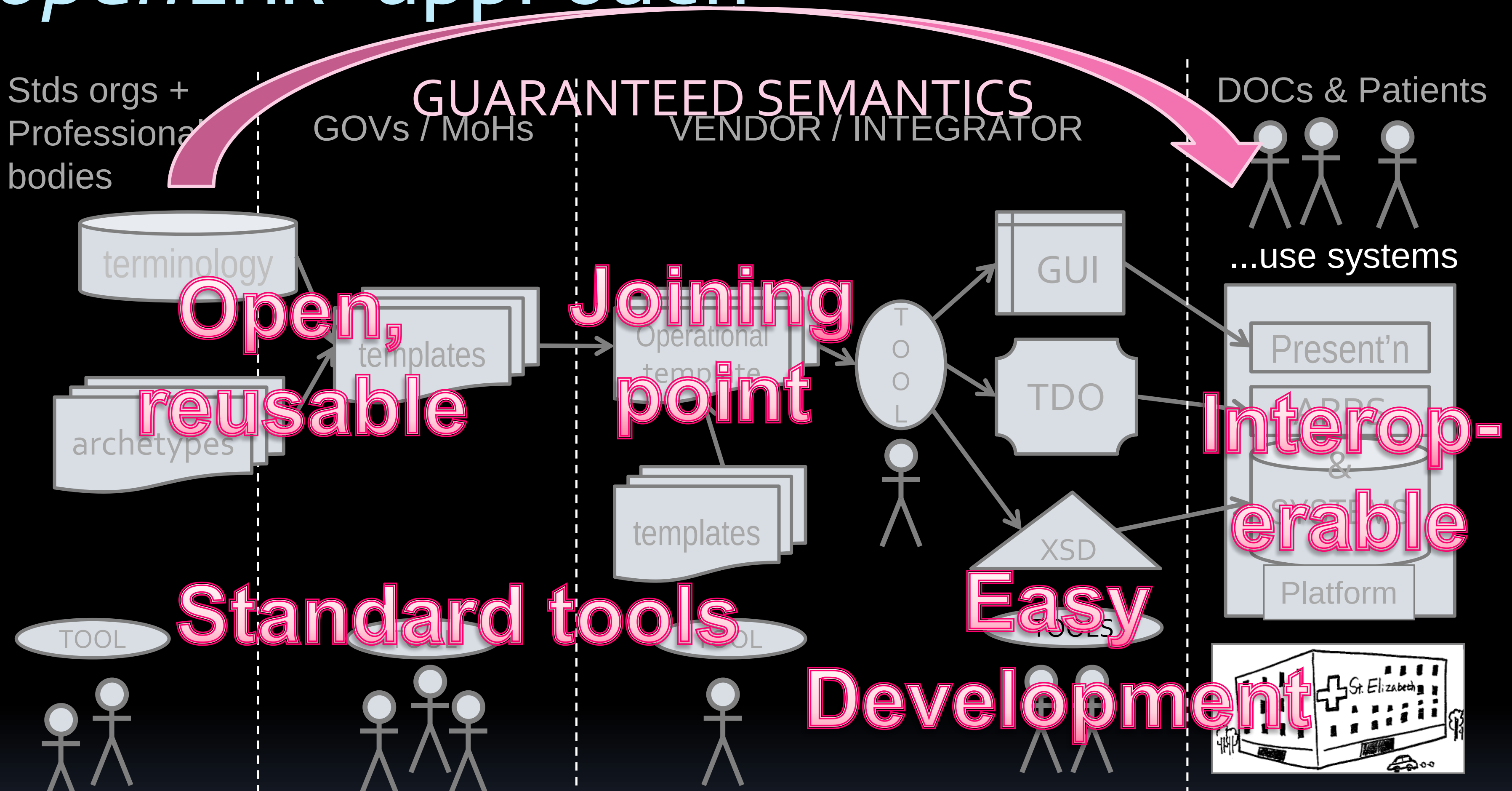
...build templates and issue as standards e.g. Discharge Summary

...consume std templates and create their own, making OPTs

developers build SOLUTIONS based on the platform

PROVIDERS buy Solutions

*open*EHR approach



build archetypes
And terminology
that define their
Information –
e.g. via IHTSDO

...build templates
and issue as
standards
e.g. Discharge
Summary

...consum
std temp
create their own,
making OPT

developers
SOLUTIONS
ased on
the platform

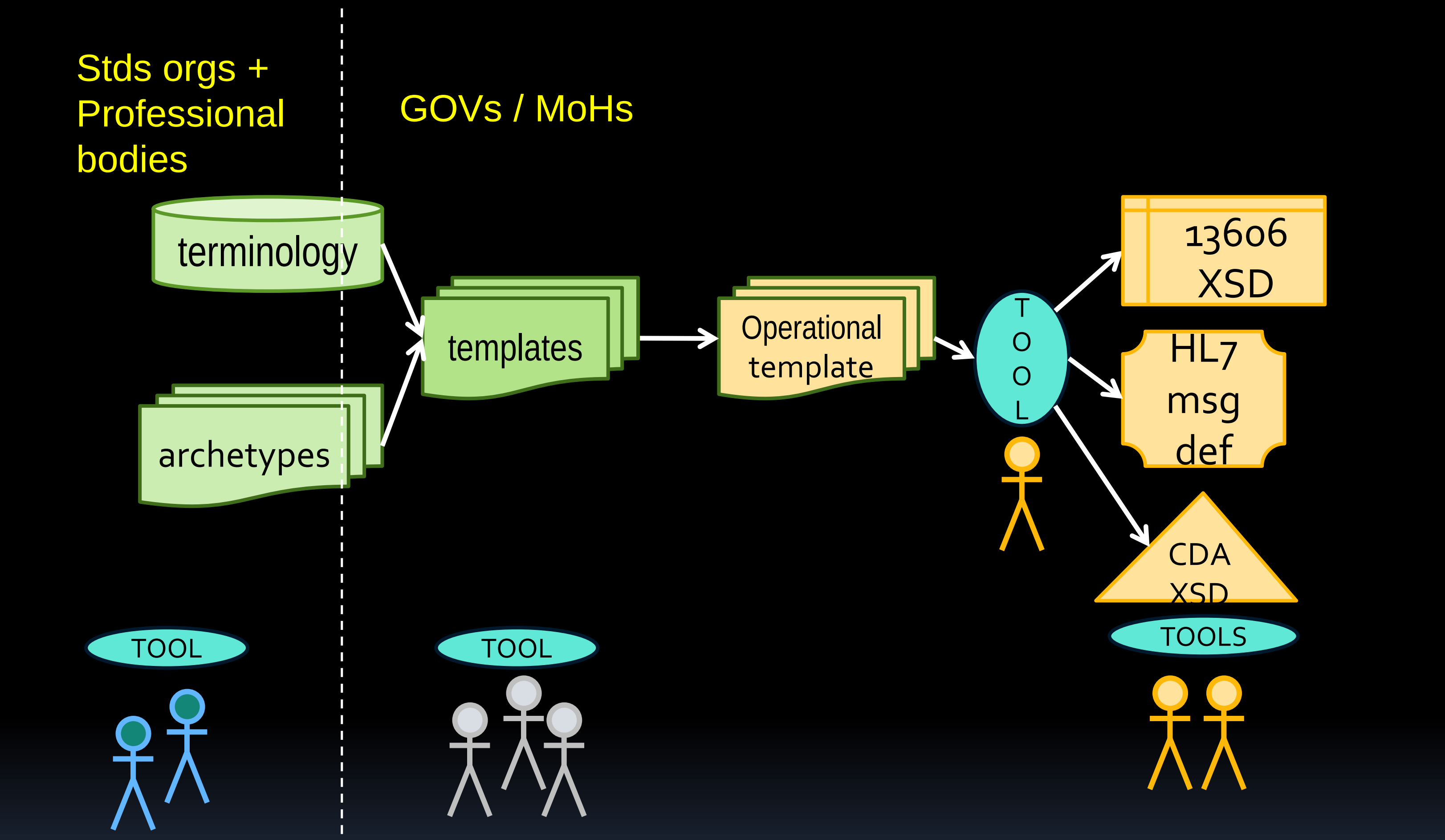
Cheaper, faster

PROVIDERS
buy
Solutions

The key....

- Is the operational template (OPT) – this is the joining point between the semantic specifications and deployable software artefacts that can be used by normal developers

Approach to standards



build archetypes & terminology that define their Information – e.g. via IHTSDO

...build templates and issue as standards e.g. Discharge Summary

Generate concrete form for dissemination to industry

Key Outcomes

- Normal developers can engage – *open*EHR + Snomed become economic and ~quick
- Semantic connection exists between definitions and implementations
  - → now we know what the meaning of data are, and DS and BI can work…
- Concrete standards like HL7 message definitions, CDA schemas, standard UI formats are DOWNSTREAM generations of operational templates

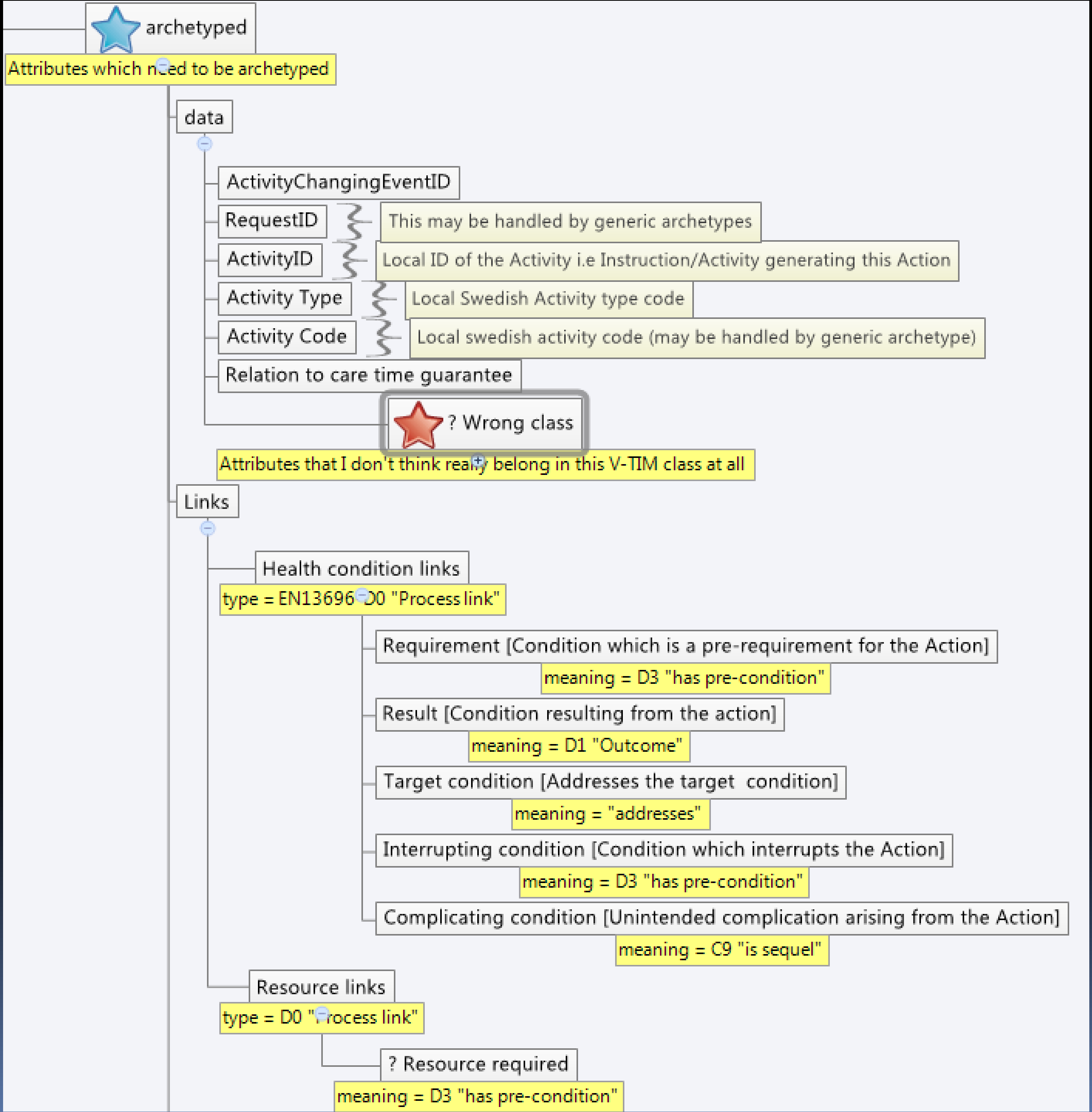
Key Requirements

- Operational Template is joining point; Tooling is key
- Portable querying language is available; also needs more tooling

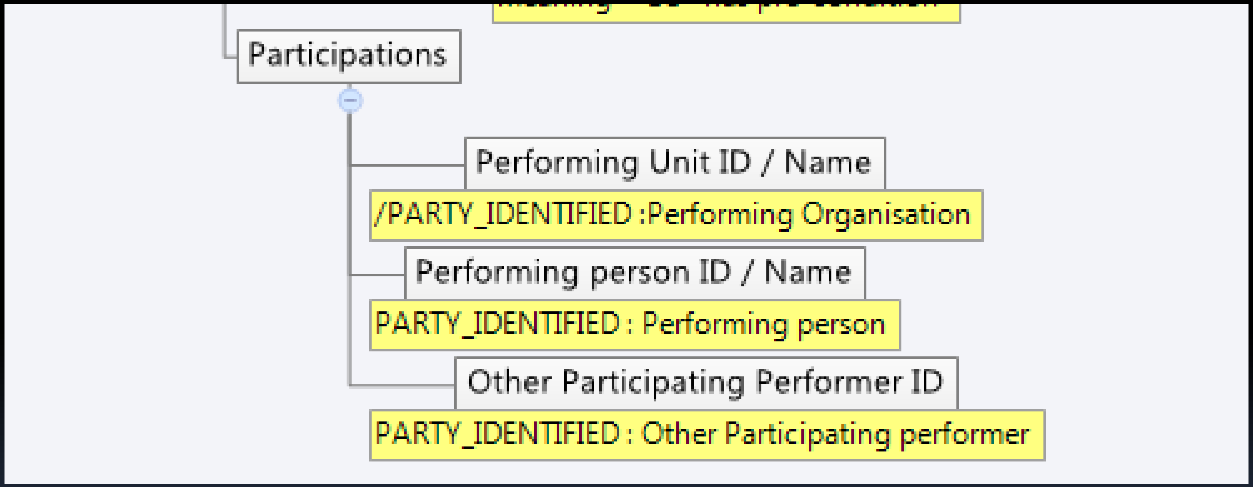
Lessons from SKL modelling

Reuse versus local authoring

- The ‘known’ needs:
  - To reuse & adapt existing archetypes (CKM, DK, …)
    - → archetypes + archetype specialisation
  - To create locally specific data-set definitions
    - → templates (in ADL 1.5: template is an archetype)
  - … So far so good
- What the SKL experience has identified:
  - The need to ‘mix-in’ locally defined but widely applicable constraints… Mostly process-related
    - This is not easy in the current archetype formalism
    - And has led to ‘reference archetypes’ approach



- archetyped
  Attributes which need to be archetyped
  - data
    - ActivityChangingEventID
    - RequestID — This may be handled by generic archetypes
    - ActivityID — Local ID of the Activity i.e Instruction/Activity generating this Action
    - Activity Type — Local Swedish Activity type code
    - Activity Code — Local swedish activity code (may be handled by generic archetype)
    - Relation to care time guarantee
    - ? Wrong class
      Attributes that I don't think really belong in this V-TIM class at all
  - Links
    - Health condition links
      type = EN13696-D0 "Process link"
      - Requirement [Condition which is a pre-requirement for the Action]
        meaning = D3 "has pre-condition"
      - Result [Condition resulting from the action]
        meaning = D1 "Outcome"
      - Target condition [Addresses the target condition]
        meaning = "addresses"
      - Interrupting condition [Condition which interrupts the Action]
        meaning = D3 "has pre-condition"
      - Complicating condition [Unintended complication arising from the Action]
        meaning = C9 "is sequel"
    - Resource links
      type = D0 "Process link"
      - ? Resource required
        meaning = D3 "has pre-condition"



Participations

Performing Unit ID / Name

/PARTY_IDENTIFIED :Performing Organisation

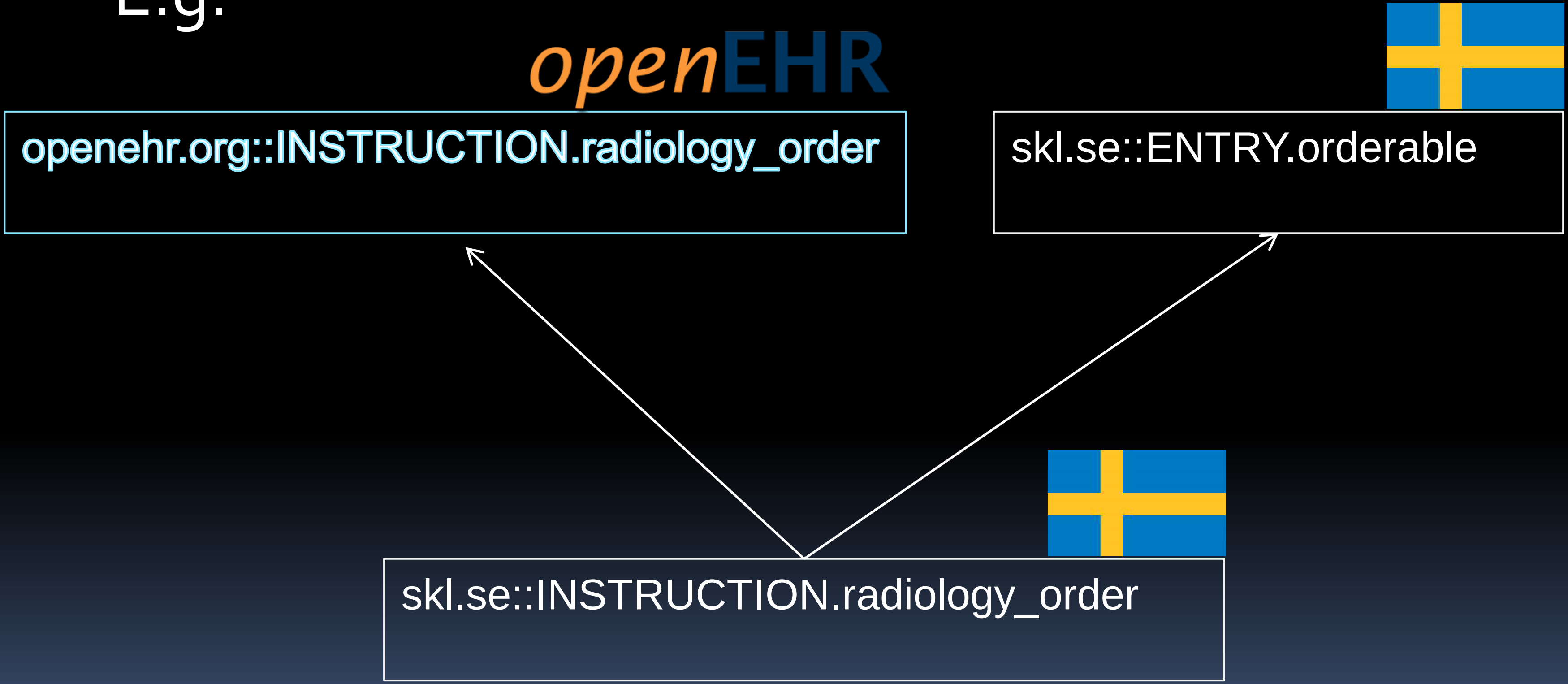
Performing person ID / Name

PARTY_IDENTIFIED : Performing person

Other Participating Performer ID

PARTY_IDENTIFIED : Other Participating performer

- The need is to define such an archetype once, and to ‘inherit’ it into e.g. All ‘orderable’ archetypes.
- E.g.


openEHR
openehr.org::INSTRUCTION.radiology_order
skl.se::ENTRY.orderable
skl.se::INSTRUCTION.radiology_order

However...

- This would require multiple inheritance
  - Initial investigations show that this would be possible, and indeed not hard, for the future
- However, not in current archetype formalism
- We don’t know enough about the information requirements

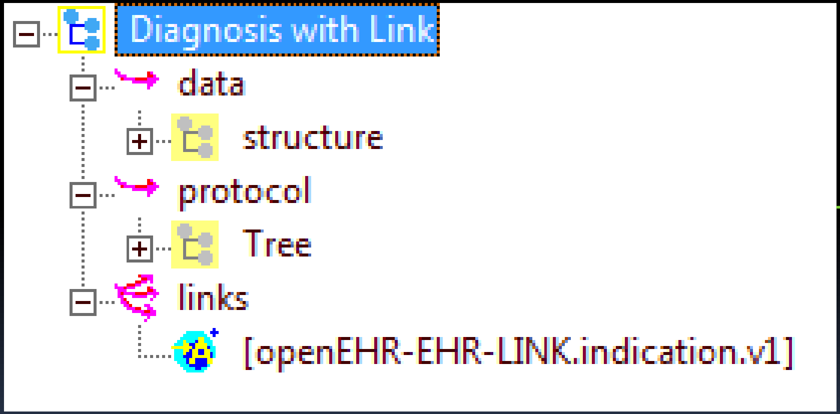
What can we do today?

1. ‘proper’ LINK archetypes to provide logical connections between e.g. OBS, Dx, Rx etc
2. Instruction index – a generalised way of tracking INSTRUCTION and ACTIONs within a Care plan structure
3. INSTRUCTION.context archetyping – recording of Request / Order ids

1. LINKs

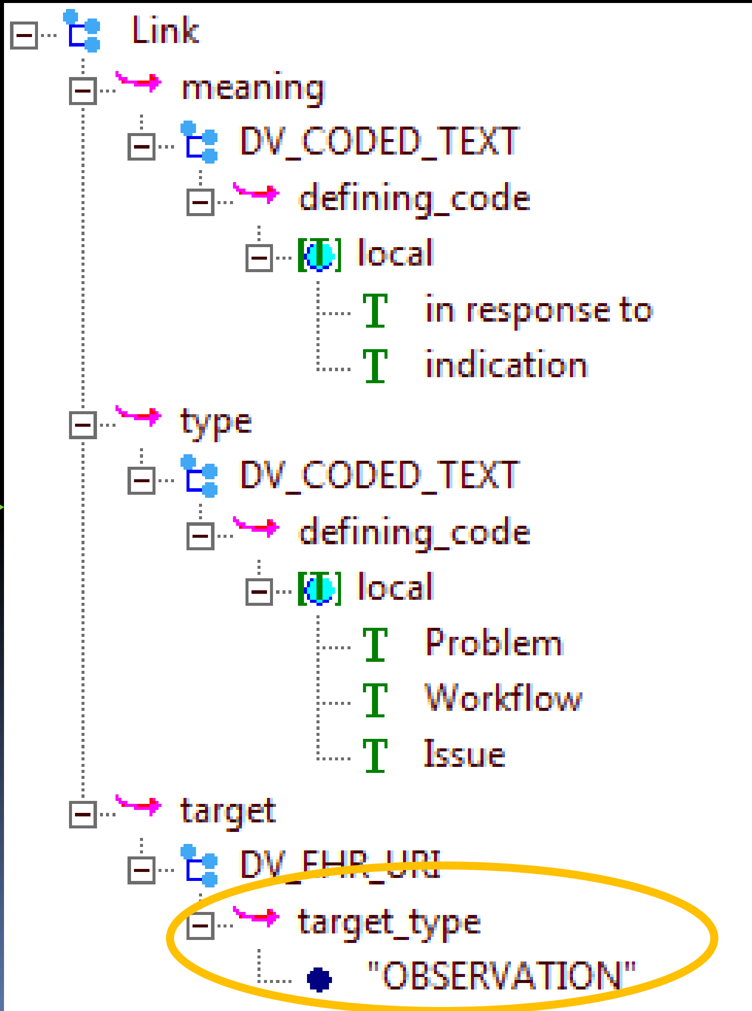
Repeat for each archetype needing SKL local constraints...

openEHR-EHR-EVALUATION .diagnosis_sweden.v1



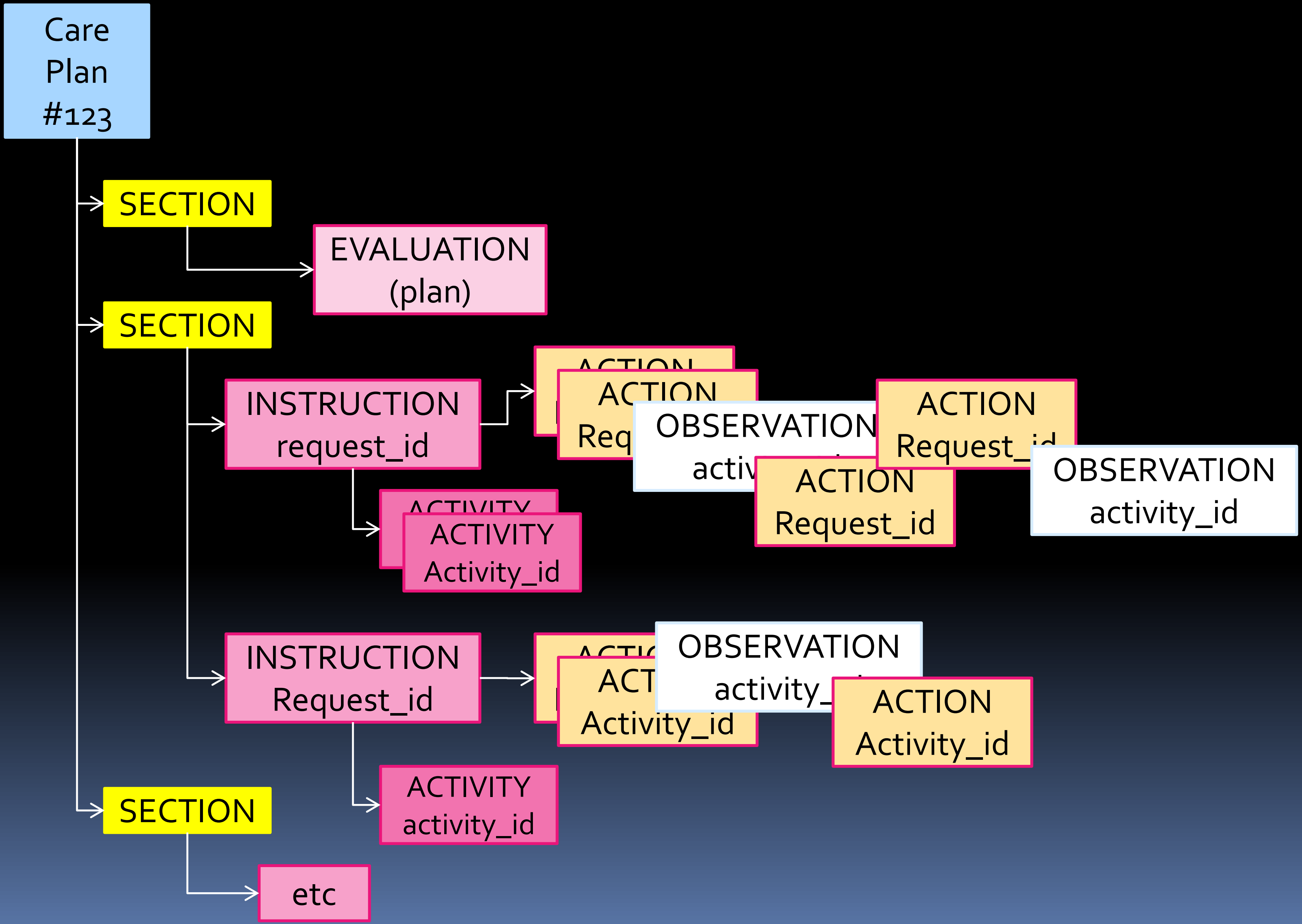
```
Diagnosis with Link
  data
    structure
  protocol
    Tree
  links
    [openEHR-EHR-LINK.indication.v1]
```

openEHR-EHR-LINK. indication.v1



```
Link
  meaning
    DV_CODED_TEXT
      defining_code
        [T] local
          T in response to
          T indication
  type
    DV_CODED_TEXT
      defining_code
        [T] local
          T Problem
          T Workflow
          T Issue
  target
    DV_EHR_URI
      target_type
        "OBSERVATION"
```

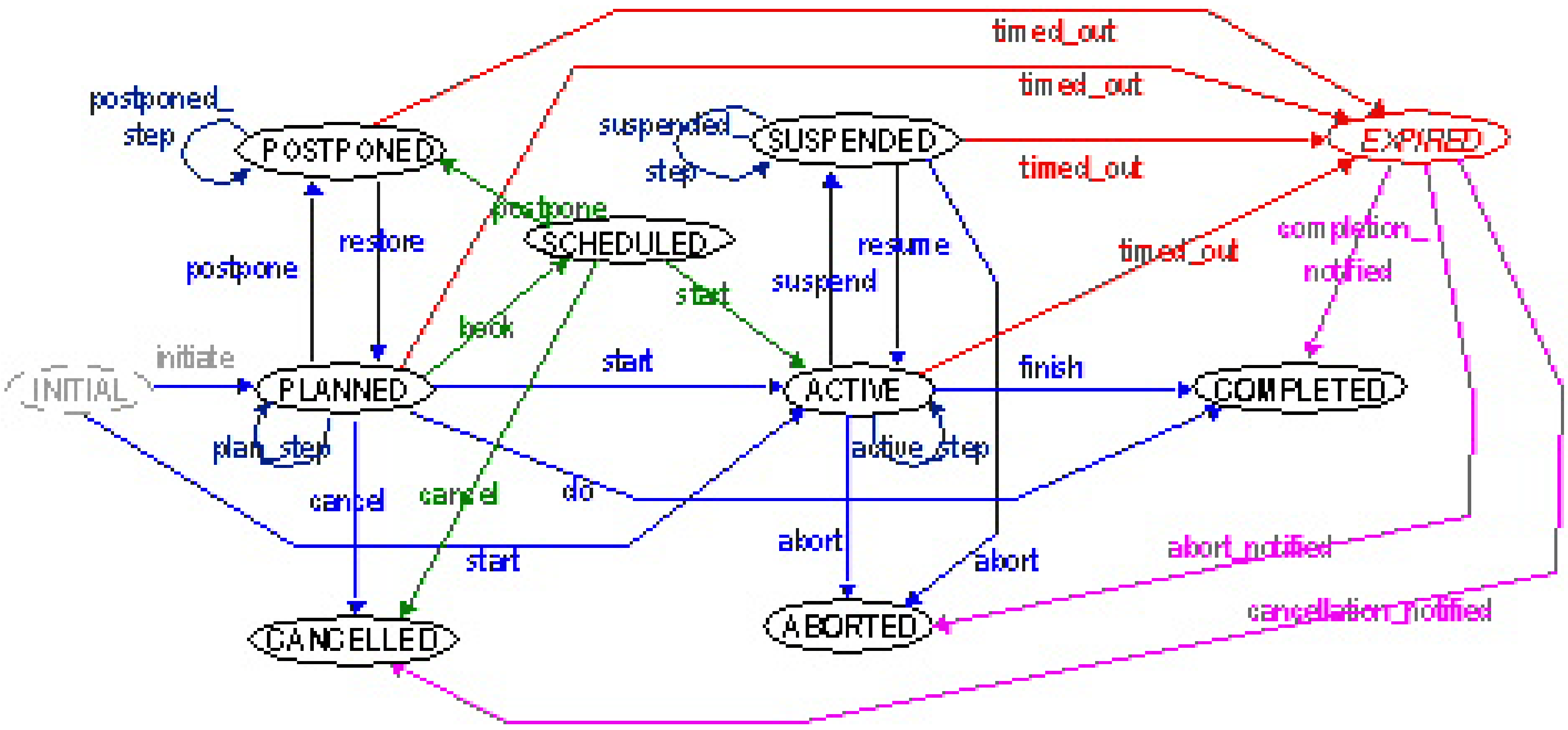
2. Instruction Index



How does it work?

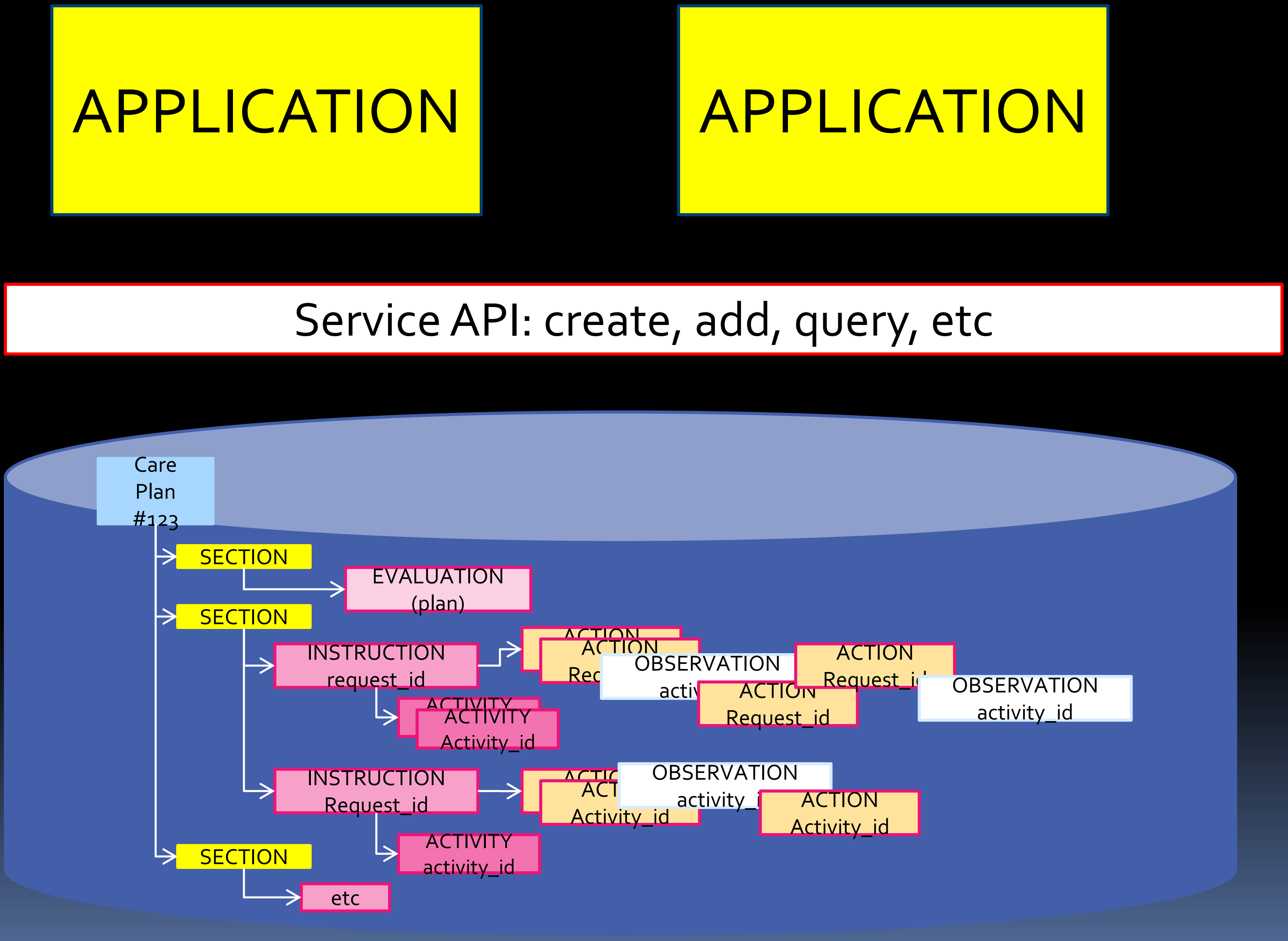
- Implemented as an *open*EHR persistent COMPOSITION
- Created by rules that match content (at least activity/request_ids)
- Can query state of any INSTRUCTION
- Can query state of whole Care Plan (algorithm to derive state based on pieces)
- Add a business Service API layer for easy use
- High performance retrieval

State machine – Instructions and Actions



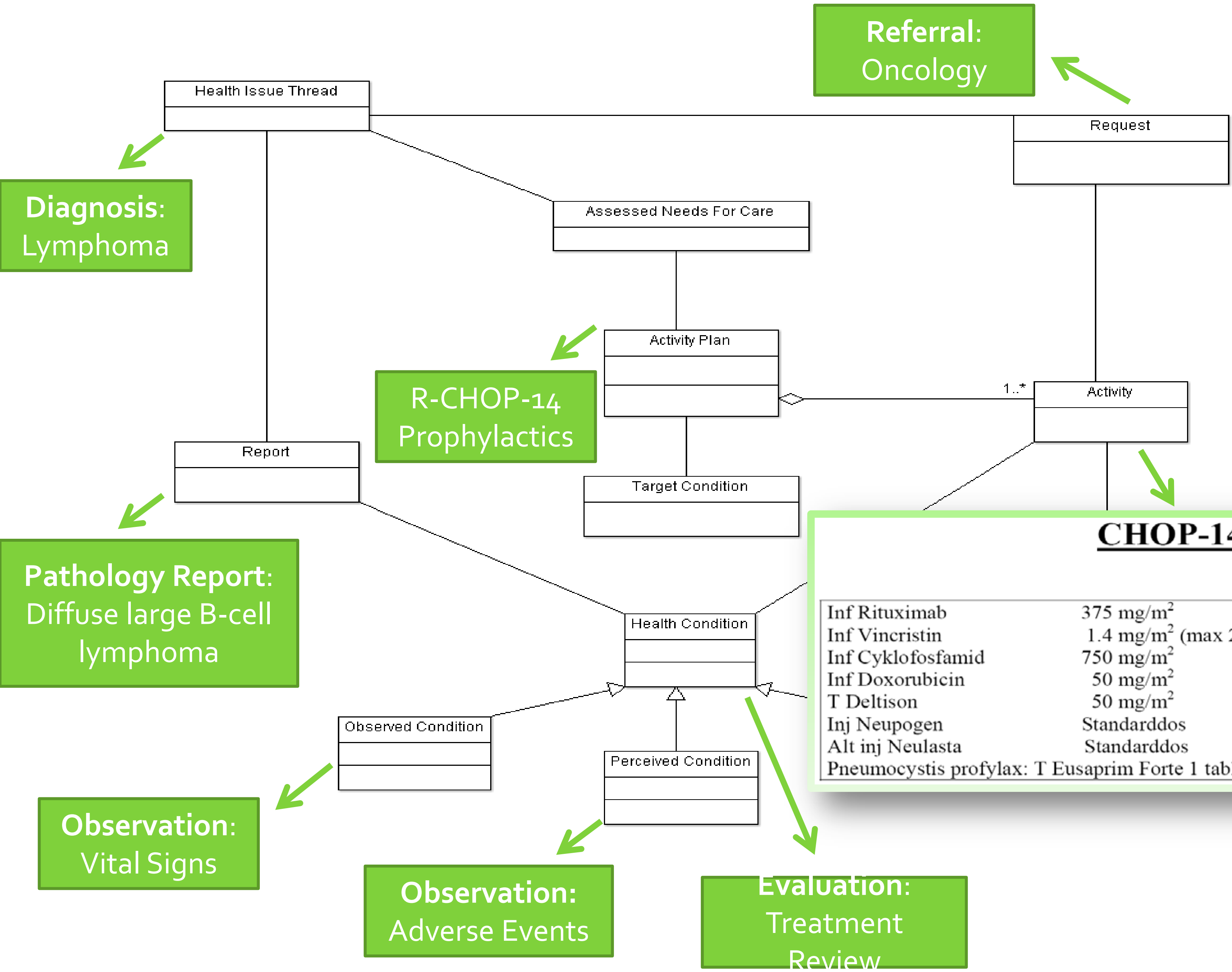
Transition legend

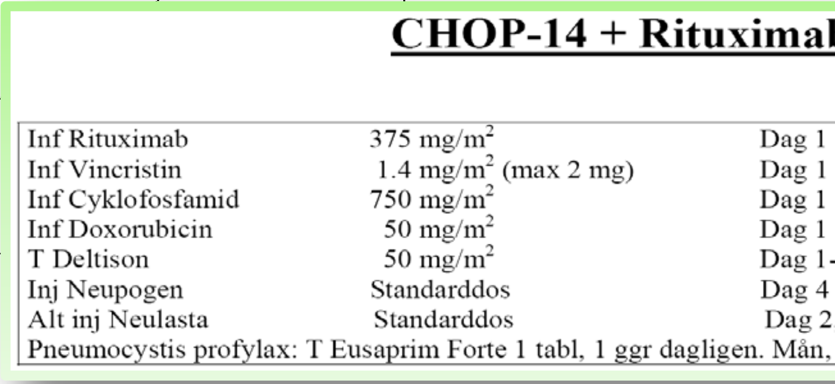
blue: normal state changing event
dark blue: normal non-state changing event
green: scheduling-related transitions
red: time-out event
magenta: post time-out event


APPLICATION
APPLICATION
Service API: create, add, query, etc
Care Plan #123
SECTION
EVALUATION (plan)
SECTION
INSTRUCTION request_id
ACTION
ACTION Rec
OBSERVATION acti
ACTION Request_id
ACTION Request_i
OBSERVATION activity_id
ACTIVITY
ACTIVITY Activity_id
INSTRUCTION Request_id
ACTIO
ACT Activity_id
OBSERVATION activity_i
ACTION Activity_id
ACTIVITY activity_id
SECTION
etc

Is it enough?

- So far, yes, in SOME circumstances
- If more than a request_id/activity_id is required, NO,
  - we need more data, either more data fields in the same Entry (needs more sophisticated archetypes)
  - AND/OR
  - We need to use LINKs to other ENTRYs
- In general, we may need to design other ENTRYs according to e.g. a MIXED process and information model




CHOP-14 + Rituximal

Inf Rituximab	375 mg/m^2	Dag 1
Inf Vincristin	1.4 mg/m^2 (max 2 mg)	Dag 1
Inf Cyklofosfamid	750 mg/m^2	Dag 1
Inf Doxorubicin	50 mg/m^2	Dag 1
T Deltison	50 mg/m^2	Dag 1-
Inj Neupogen	Standarddos	Dag 4
Alt inj Neulasta	Standarddos	Dag 2

Pneumocystis profylax: T Eusaprim Forte 1 tabl, 1 ggr dagligen. Mån,

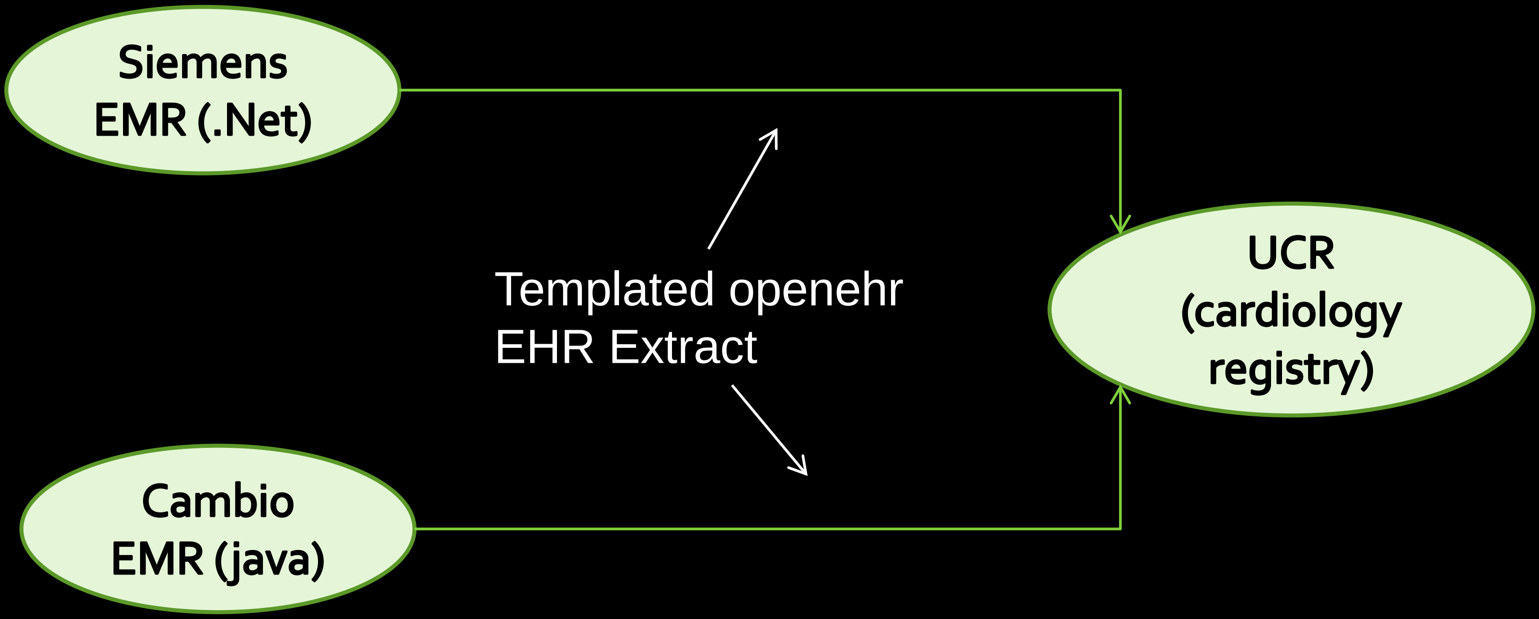
Stepping back: more analysis is needed

- What eventual information structures are required?
- What links / indexing is required?
- What querying is required…
- What is the ‘model of use’?
  - Is it Entries with multiple ‘flavours of information’?
  - Is it Clinical Entries LINKed with process Entries with process engine behind?
  - Is it some kind of indexing structure?
  - Is it a combination?

- CKM archetypes implement globally typical clinical & demographic content
- SKL’s ‘general content’ is nearly all process-related
- Other experience in e.g. Slovenia shows need for ‘nursing view’, i.e. Task list: todo/done/…
- ➔ at least 3 dimensions
  - (Clinical) information
  - Process / orders
  - Tasks / work done

Lessons from IFK2

Requirement



Template:
- Swedish language;
- some SNOMED CT coding;
- 100 data points

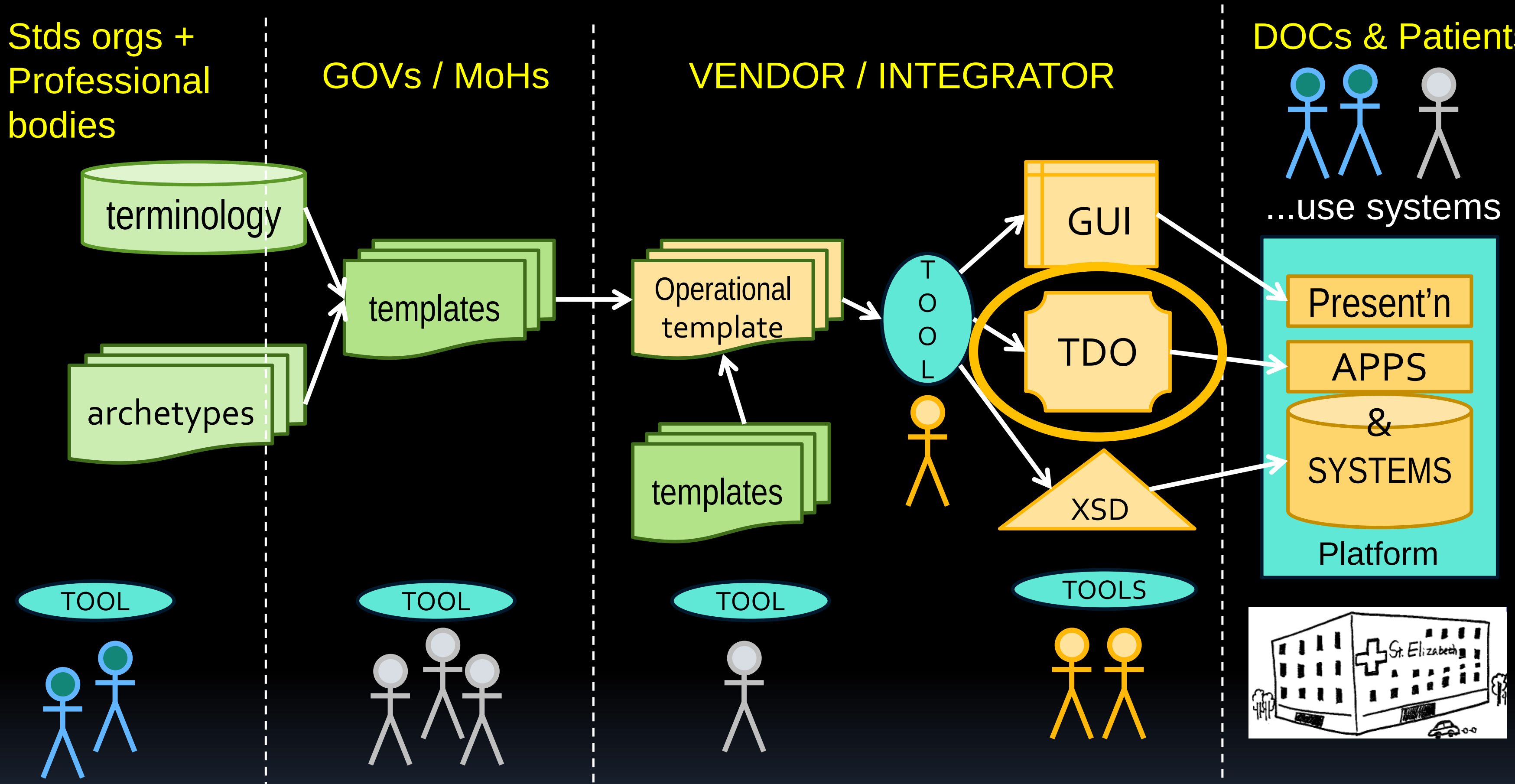
The content

- About 75 archetypes in the IFK2- RiksVikt Secondary Care template
- About 66 in the IFK2- RIKSVIkt primary care template

What we learned...

- It was somewhat painful – various details were incorrect
  - Now both .Net and Java generated Extracts accepted at UCR end
- ‘TDO’ approach works
- TDO spec will be offered to *open*EHR.org
- Generalised approach: generate TDO from template; calls to TDO will generate correct data, e.g. EHR Extract

*open*EHR approach



build archetypes & terminology that define their Information – e.g. via IHTSDO

...build templates and issue as standards e.g. Discharge Summary

...consume std templates and create their own, making OPTs

developers build SOLUTIONS based on the platform

PROVIDERS buy Solutions

*open*EHR 2.0 – some ideas

General concept

- The EHR is no longer just clinical, but a generalised health information resource
- ‘Clinical’ EHR is one view
- Clinicians need to *choose* what data needs to go into permanent lists
- → don’t put e.g. Task tracking in own DB
- → need some performance improvements
- → need active rules
- → need notification capability

Entry index

- Already in use in Australia
- As Entries are created, rules are invoked to find matches, e.g. Any lab result containing ‘MRSA’ (flesh-eating bug)
- If fired, add this content (or a ref) to a fast ‘index’ Composition with the EHR
- Entry indexes can be created, destroyed, recreated
- Good for time-based data being graphed
- Spec will be offered to *open*EHR Q1 2011

Archetype multiple inheritance?

- Deal with multiple ‘dimensions’, e.g.
  - Information
  - Process / orders
  - Tasks
- More research needed

Order support in RM

- Addition of order / filler ids
  - Needs to be done carefully
  - Needs to handle multiple ids
  - Messy real world situations

Task list support

- Assignation of task
- Record completion or step
- Overdue alerts
- Notification of task complete

Careplan support

- Careplan is a kind of Entry Index, with other features
- See work of Rong Chen et al.