HD3: Hazard driven Decomposition, Design and Development

Driving design by hazard analysis

Nicholas Mc Guire <safety@osadl.org>
Markus Kreidl <mkreidl@opentech.at>

August 31, 2025

Motivation

- Growing complexity -> mandates maximized context
- Refocus on hazard elimination
- Heavy use of pre-existing elements
- Maintenance - expecting high dynamics during life-time

This is using many ideas from other methods and is much more of an consolidation attempt in design context than a new method in its own right.

Nicholas Mc Guire <safety@osadl.
Markus Kreidl <mkreidl@oper

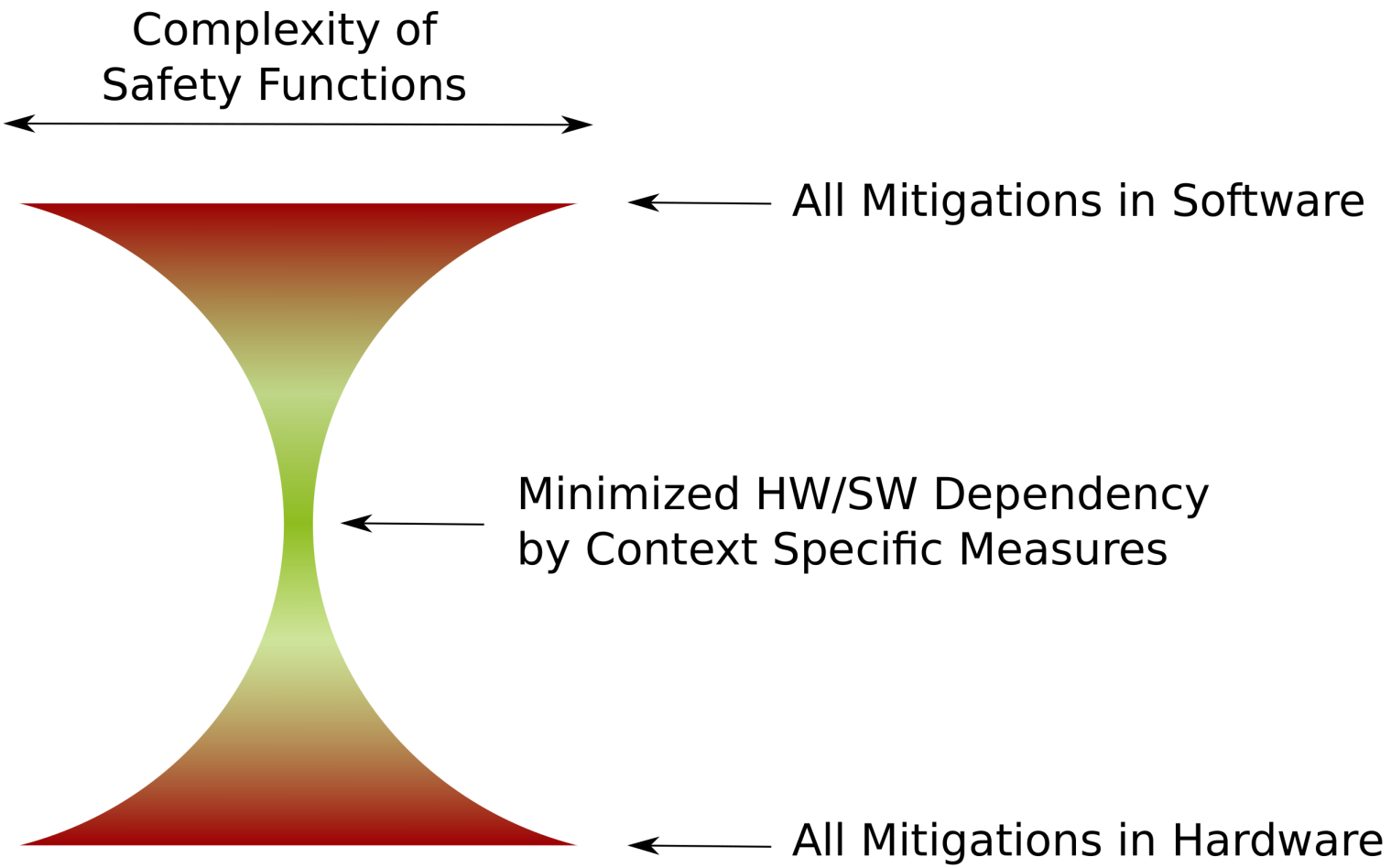
Guiding Principle

> *As far as practicable the design shall keep the safety-related part of the software simple.*
>
> [IEC 61508-3 Ed 2 7.4.2.6]

Thus using complex elements safely is the goal rather than making the complex elements safe !

Nicholas Mc Guire <safety@osadl.
Markus Kreidl <mkreidl@oper

Context

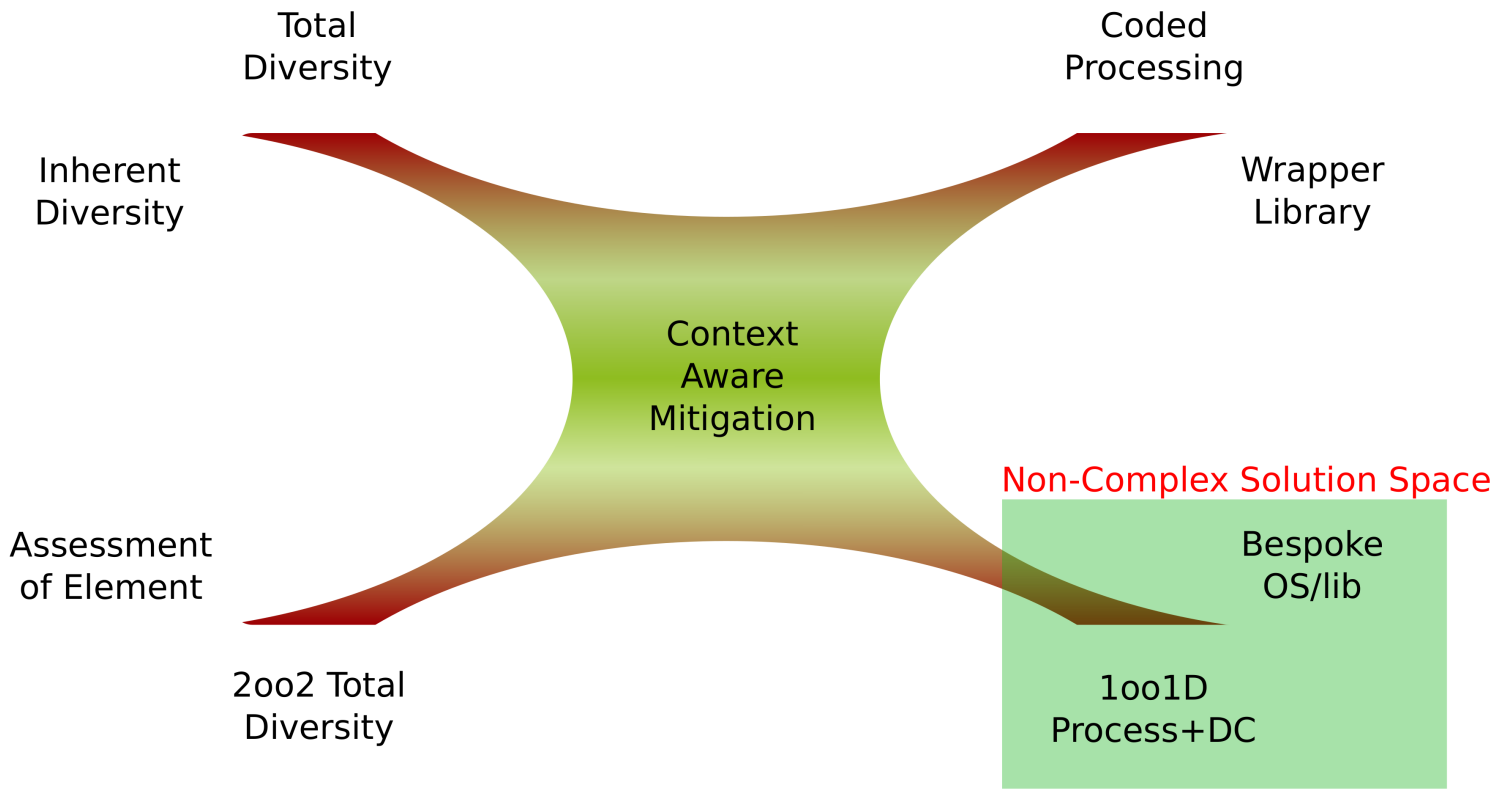


HD3: Hazard driven Decomposition, Design and Development

Driving design by hazard analysis

Nicholas Mc Guire
<safety@osadl.
Markus Kreidl
<mkreidl@oper

Complex Systems - Solution Space



HD3: Hazard driven Decomposition, Design and Development

Driving design by hazard analysis

Nicholas Mc Guire
<safety@osadl.
Markus Kreidl
<mkreidl@oper

Needs in Software Intensive Systems

- Exploratory methodology needed - FMEA/FTA will not do
- Drive the design by hazards and not functionality to
  - Develop sufficient specific analytical context
  - Refocus on hazard elimination opportunities
  - Minimize complexity of safety functions
  - Maximize maintainability and support impact analysis
- Create a complete traceable hierarchy of safety related functions supporting defense in depth

You want a minimized “hazard-surface” and provide a fully understood hazard dependency or your impact analysis will be prohibitive.

Driving design by hazard analysis

Nicholas Mc Guire <safety@osadl.
Markus Kreidl <mkreidl@oper

Goal

- Eliminating as many hazards along the way as possible by treating design as emergent
- Allocating suitable mitigations at multiple levels where necessary
- Developing and recording the entire hazard hierarchy to ensure maintainability of system-safety properties during modification

Our expectation is that in complex software centric safety related systems modification and retrofitting will by far more common than in traditional safety-related systems - provisions for this paradigm change we think are critical.

Nicholas Mc Guire <safety@osadl.
Markus Kreidl <mkreidl@oper

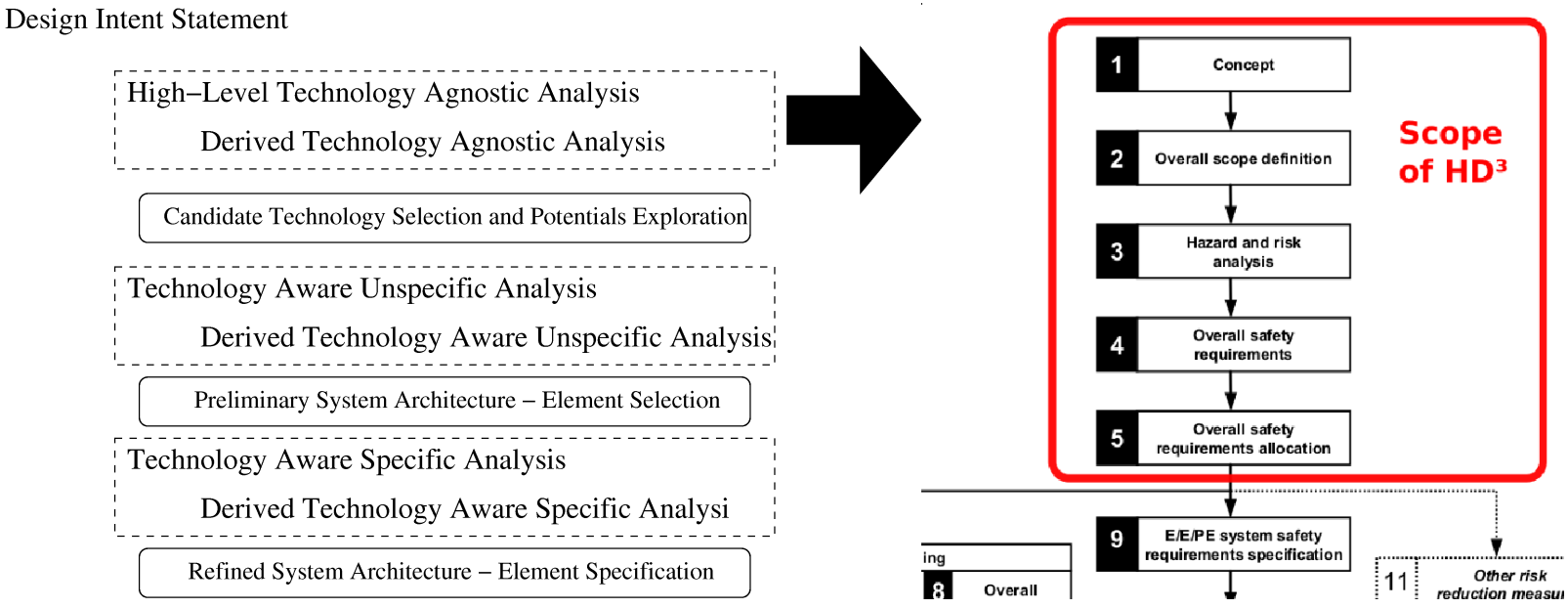
Starting Point - Design Intent

Nicholas Mc Guire <safety@osadl.
Markus Kreidl <mkreidl@oper

> *Measure and report E.Coli specific enzymatic activity per volume of sample (indicating the level of fecal contamination) in at most 15 minutes, with defined and verifiable level of assurance.*
>
> [Coliminder Design Intent]

- V-Model: Requirements -> Design -> implement ...
- HD3: Intent -> Hazards -> Mitigations -> Elements

HD3 Method Overview



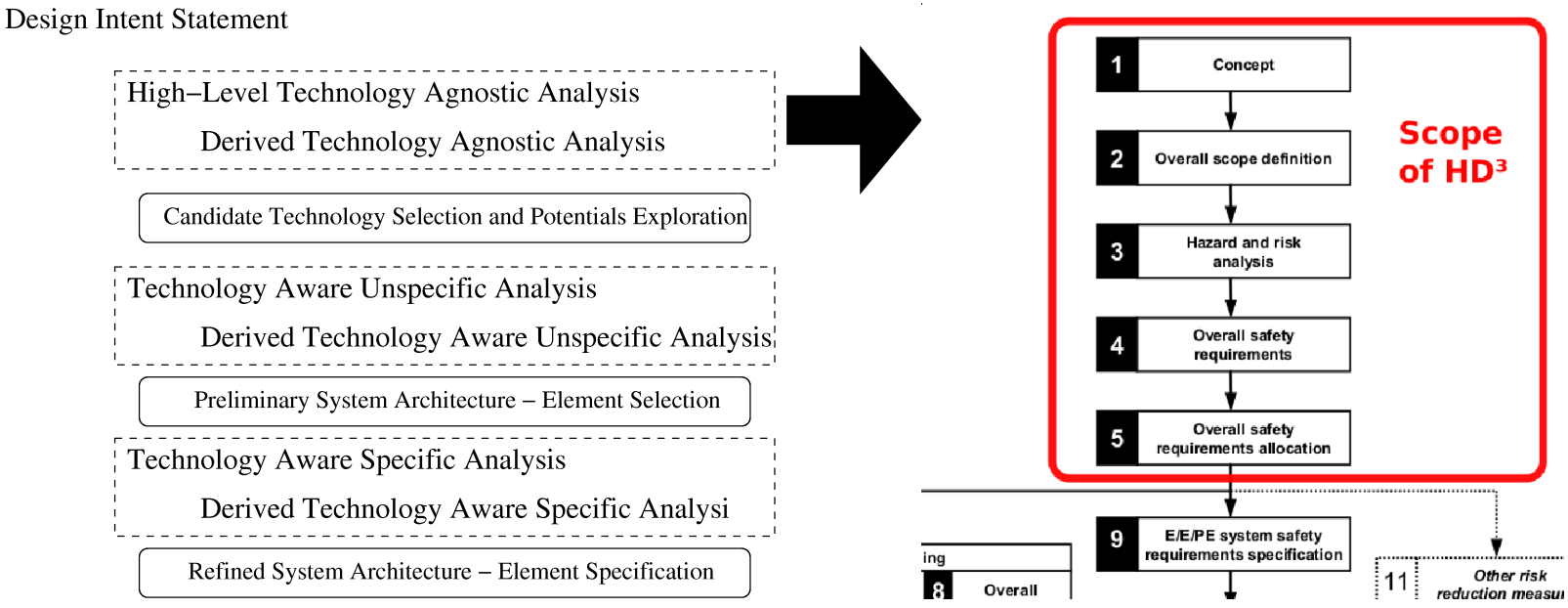
HD3: Hazard driven Decomposition, Design and Development

Driving design by hazard analysis

Nicholas Mc Guire <safety@osadl.

Markus Kreidl <mkreidl@oper

HD3 Method Overview



- Technology Agnostic Layer -> IEC 61508 Ed 2 Part 1
- Technology Aware Unspecific -> roughly IEC 61508 Ed 2 Part 2/3 requirements
- Technology Aware Specific -> roughly IEC 61508 Ed 2 Part 2/3 design

HD3: Hazard driven Decomposition, Design and Development

Driving design by hazard analysis

Nicholas Mc Guire <safety@osadl.
Markus Kreidl <mkreidl@oper

HD3 High-level Example

Interpretation->Cause

```
# Item: run_NTask
## Origin: [TOP_Level](#top-level)
## Guideword: Other Than

## Interpretation
1. Spurious CTask activated
2. Spurious halt initialized
3. Wrong Task set provided

## Cause
1. Spurious CTask activated
  1. Faulty startup
  2. NTasks handling has access to CTasks
2. Spurious halt initialized
  1. NTask has access to halt (system control)
  2. Side-effect of NTask (shared resource)
3. Wrong Task set provided
  1. NTasks not properly managed
  2. NTasks not well selected
  3. Unauthorized access to system
...
```


Nicholas Mc Guire <safety@osadl.
Markus Kreidl <mkreidl@oper

HD3 High-level Example - cont.

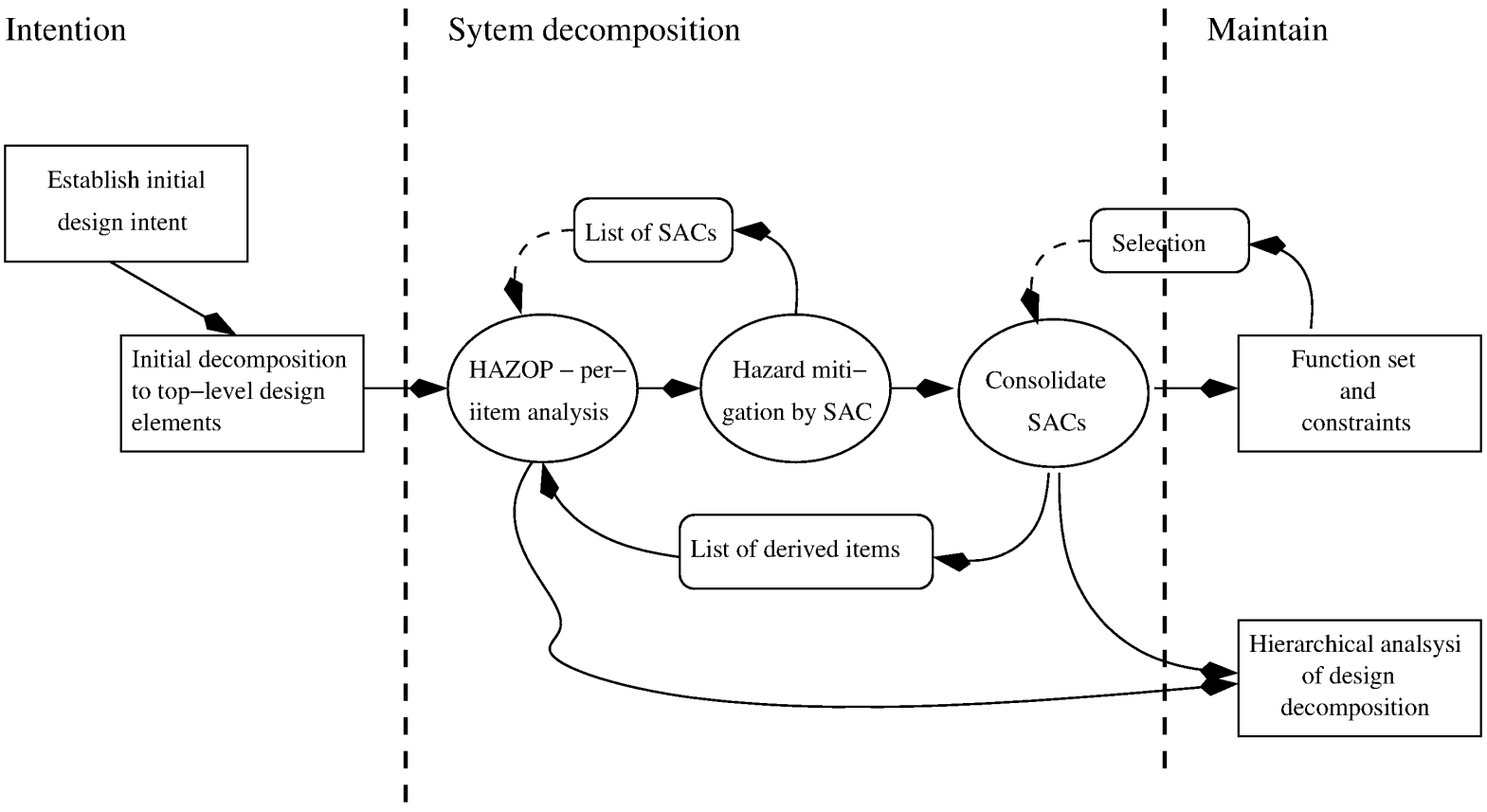
Consequence->Mitigation

```
## Mitigation
1. Spurious CTask activated
  1. Faulty startup
    - Unexpected side-effect of configured valid resoruce
       * [HLS_3](#configuration-verificaton-failure-transits-to-halt))
       * [HLS_7](#task-set-verification-failure-transits-to-halt)
       * [HLS_8](#All-CTasks-instantiated-prior-to-NTasks)
       * [HLS_30](#CTask-configurations-include-unique-ID)
       * [HLS_36](#Runtime-resource-isolation)
       * [HLS_54](#CTask-shares-isolated-resources-with-CTasks-only)
       * [HLS_99](#Managed-deployment-of-all-system-elements)
       * [HLS_100](#Managed-selection-of-all-system-elements)
       * [HLS_101](#Access-control-violation-terminates-NTask)
    - Conflicting actions
       * [HLS_3](#configuration-verificaton-failure-transits-to-halt))
       * [HLS_7](#task-set-verification-failure-transits-to-halt)
       * [HLS_8](#All-CTasks-instantiated-prior-to-NTasks)
       * [HLS_36](#Runtime-resource-isolation)
       * [HLS_54](#CTask-shares-isolated-resources-with-CTasks-only)
       * [HLS_99](#Managed-deployment-of-all-system-elements)
       * [HLS_100](#Managed-selection-of-all-system-elements)
       * [HLS_101](#Access-control-violation-terminates-NTask)
    - Conflicting actions
```


Driving design by hazard analysis

Nicholas Mc Guire <safety@osadl.
Markus Kreidl <mkreidl@oper

HD3 Flow

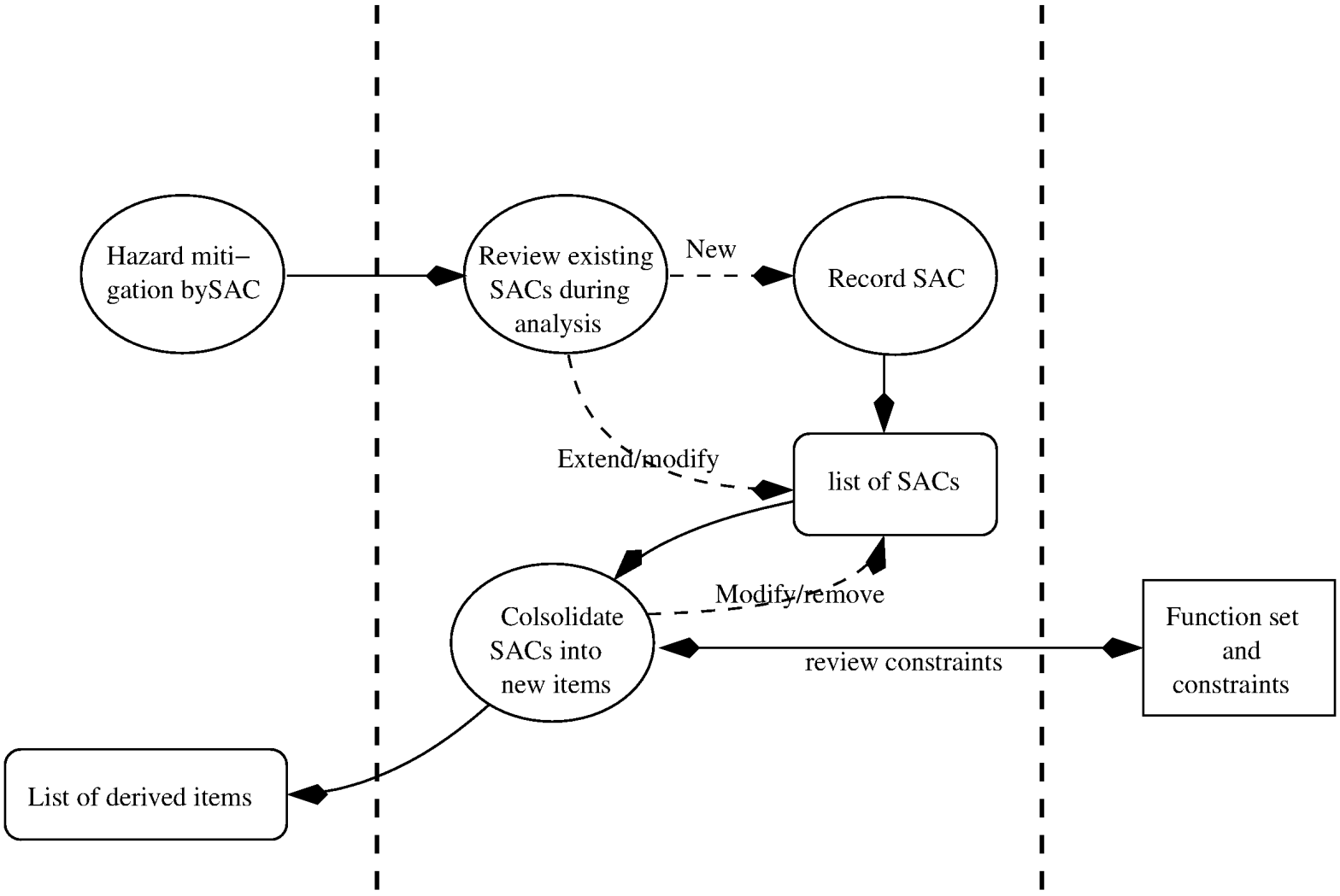


HD3: Hazard driven Decomposition, Design and Development

Driving design by hazard analysis

Nicholas Mc Guire <safety@osadl.
Markus Kreidl <mkreidl@oper

HD3 Continuous Consolidation



Driving design by hazard analysis

Nicholas Mc Guire <safety@osadl.
Markus Kreidl <mkreidl@oper

SAC development -> Requirements

1. List of undeveloped SACs (synopsis only)
2. Consolidate SACS -> merge “obviously” related SACs
3. List of SAC call sites -> context
4. List of cases to protect against (causes) -> functional “specification”
5. For each SAC develop full description
6. Validate description against context (list of call-sites) -> safety specification
7. If consistent specification not feasible -> split SAC -> update context

SAC description is then input to the protection code specification/implementation

Driving design by hazard analysis

Nicholas Mc Guire <safety@osadl.
Markus Kreidl <mkreidl@oper

Conclusion

- High-complexity systems need a minimizing design methodology with a focus on safety or complexity is unmanageable
- We need to regain the potentials for hazard elimination - functionally driven methods encourage mitigation !
- HD^3 is experimental - first results are encouraging - if it really works - we do not yet know
- Next step: find a mid complexity system and run HD^3 - e.g. lane assistant ...

Nicholas Mc Guire <safety@osadl.
Markus Kreidl <mkreidl@oper

Nicholas Mc Guire <safety@osadl.
Markus Kreidl <mkreidl@oper

Thanks !

http://www.osadl.org/SIL2
safety@osadl.org