A minimalist formal framework for systems architecting

IWMBSA’13

Boris Golden, Yann Hourdel

March 25, 2013
1. Goals
2. Preliminary definitions
3. Systemic behaviour specification
4. Systems as recursive structures
Goals

Preliminary definitions
Systemic behaviour specification
Systems as recursive structures
Conclusion

1 Goals
Start point

Goals
Preliminary definitions
Systemic behaviour specification
Systems as recursive structures
Conclusion

There exists (at least) one pretty good architecture framework.

How to put formal models behind it?

How to compute/propagate properties on such models?

How to prove coherence between systemic layers?
There exists (at least) one pretty good architecture framework.
There exists (at least) one pretty good architecture framework.

- How to put formal models behind it?
Start point

There exists (at least) one pretty good architecture framework.

- How to put formal models behind it?
- How to compute/propagate properties on such models?
Start point

There exists (at least) one pretty good architecture framework.

- How to put formal models behind it?
- How to compute/propagate properties on such models?
- How put prove coherence between systemic layers?
Objectives

Goals

Preliminary definitions

Systemic behaviour specification

Systems as recursive structures

Conclusion

We will try to:

- define a coherent set of formal notions to describe models using views
- give informal ideas on how to use them in a framework
- give insight on how to study (safety) properties on such models

But we won't:

- provide a tool implementing such notions
- provide a mathematical environment for properties propagation
Objectives

We will try to:

- define a coherent set of formal notions to describe models using views
- give informal ideas on how to use them in a framework
- give insight on how to study (safety) properties on such models

But we won't:

- provide a tool implementing such notions
- provide a mathematical environment for properties propagation
Objectives

We will try to:

- define a coherent set of formal notions to describe models using **views**
Objectives

We will try to:

- define a coherent set of formal notions to describe models using **views**
- give informal ideas on how to use them in a framework
Objectives

We will try to:

- define a coherent set of formal notions to describe models using **views**
- give informal ideas on how to use them in a framework
- give intel on how to study (safety) properties on such models
Objectives

We will try to:

- define a coherent set of formal notions to describe models using **views**
- give informal ideas on how to use them in a framework
- give intel on how to study (safety) properties on such models

But we won’t:
Objectives

We will try to:

- define a coherent set of formal notions to describe models using views
- give informal ideas on how to use them in a framework
- give intel on how to study (safety) properties on such models

But we won’t:

- provide a tool implementing such notions
Objectives

We will try to:

- define a coherent set of formal notions to describe models using **views**
- give informal ideas on how to use them in a framework
- give intel on how to study (safety) properties on such models

But we won’t:

- provide a tool implementing such notions
- provide a mathematical environment for properties propagation
About systemic views

Goals
Preliminary definitions
Systemic behaviour specification
Systems as recursive structures
Conclusion

We need to make coherent a set of:
- functional views
- constructional views
- safety views
...
We need to make coherent a set of:

- functional views
- constructional views
- safety views

...
We need to make coherent a set of:

- functional views
About systemic views

We need to make coherent a set of:

- functional views
- constructional views
About systemic views

We need to make coherent a set of:

- functional views
- constructional views
- safety views
- ...
2 Preliminary definitions
Time & data

Key points:
- Unified definition for continuous and discrete time scales
Time & data

Key points:

- Unified definition for continuous and discrete time scales
- Datasets come with their own read/write behaviour
Time & data

Key points:

- Unified definition for continuous and discrete time scales
- Datasets come with their own read/write behaviour
- Dataflows are datasets carried by time scales
Black system

A **black system** is a 7-tuple $S = (\mathbb{T}, X, Y, Q, q_0, \mathcal{F}, \delta)$ where:

- $\mathbb{T}$ is a time scale,
- $X$, $Y$ are *input* and *output* datasets,
- $Q$ is a nonempty $\varepsilon$-alphabet of *states*,
- $q_0$ is an element of $Q$, called *initial state*,
- $\mathcal{F} : X \times Q \times \mathbb{T} \rightarrow Y$ describes a *functional behavior*,
- $\delta : X \times Q \times \mathbb{T} \rightarrow Q$ describes a *state behavior*. 
A **black system** is a 7-tuple $S = (\mathbb{T}, X, Y, Q, q_0, \mathcal{F}, \delta)$ where:

- $\mathbb{T}$ is a time scale,
- $X$, $Y$ are *input* and *output* datasets,
- $Q$ is a nonempty $\varepsilon$-alphabet of states,
- $q_0$ is an element of $Q$, called *initial state*,
- $\mathcal{F} : X \times Q \times \mathbb{T} \rightarrow Y$ describes a *functional behavior*,
- $\delta : X \times Q \times \mathbb{T} \rightarrow Q$ describes a *state behavior*. 

![Diagram of black system](attachment:diagram.png)
Operations on black systems

What we can do:

- execute a system
Operations on black systems

What we can do:

- execute a system
- extend a system to a greater time scale
Operations on black systems

What we can do:

- execute a system
- extend a system to a greater time scale
- compose the product of several systems
Operations on black systems

What we can do:

- execute a system
- extend a system to a greater time scale
- compose the product of several systems
- put a feedback loop on a system
Operations on black systems

What we can do:

- execute a system
- extend a system to a greater time scale
- compose the product of several systems
- put a feedback loop on a system
- abstract / concretize a system
3 Systemic behaviour specification
A **systemic signature** is a 4-tuple \((X, Y, Q, \mathbb{T})\) where:

- \(X\), \(Y\) and \(Q\) are datasets (respectively called *input values*, *output values* and *states*),
- \(\mathbb{T}\) is a time scale.
A **systemic signature** is a 4-tuple \((X, Y, Q, T)\) where:

- \(X\), \(Y\) and \(Q\) are datasets (respectively called *input values*, *output values* and *states*),
- \(T\) is a time scale.
• Let $\Sigma = (X, Y, Q, T)$ be a systemic signature.

A requirement on $\Sigma$, is a logical formula expressing properties on the behavior of any black system of systemic signature $\Sigma$. 
Let $\Sigma = (X, Y, Q, T)$ be a systemic signature.

A requirement on $\Sigma$, is a logical formula expressing properties on the behavior of any black system of systemic signature $\Sigma$.

The set of all requirements on this systemic signature is noted $Req(X, Y, Q, T)$ or $Req(\Sigma)$.
Black box

A **black box** is a 5-uplet \((X, Y, Q, T, r)\) where:

- \((X, Y, Q, T)\) is a systemic signature
- \(r \in \text{Req}(X, Y, Q, T)\)
A **black box** is a 5-uplet \((X, Y, Q, T, r)\) where:

- \((X, Y, Q, T)\) is a systemic signature
- \(r \in Req(X, Y, Q, T)\)
Realization of a black box

Let $B = (X, Y, Q, \mathbb{T}, r)$ be a black box.

A **realization** of $B$ is any black system $S$ of systemic signature $(X, Y, Q, \mathbb{T})$ such that $S \models r$. 
Realization of a black box

Let $B = (X, Y, Q, T, r)$ be a black box.

A realization of $B$ is any black system $S$ of systemic signature $(X, Y, Q, T)$ such that $S \models r$.

When such a black system exists, $B$ is said to be realizable.
Let $B = (X, Y, Q, T, r)$ be a black box.

A realization of $B$ is any black system $S$ of systemic signature $(X, Y, Q, T)$ such that $S \models r$.

When such a black system exists, $B$ is said to be \textit{realizable}.
Concretization of a black box (1/2)

• Let $B_c \in BB(X_c, Y_c, Q_c, \mathbb{T}_c)$ (called concrete black box).
• Let $B_a \in BB(X_a, Y_a, Q_a, \mathbb{T}_a)$ (called abstract black box).
• Let $\alpha : (X_c, Y_c, Q_c, \mathbb{T}_c) \rightarrow (X_a, Y_a, Q_a, \mathbb{T}_a)$ be an abstraction mechanism.

We say that $B_c$ concretizes $B_a$ via $\alpha$ if and only if: for any black system $S_c$ that is a realization of $B_c$, $\alpha(S_c)$ is a realization of $B_a$. 
Concretization of a black box (2/2)

\[
\begin{align*}
T_c & \quad Q_c(t) \\
X_c(t) & \quad Y_c(t)
\end{align*}
\]

\[
\alpha \left( \begin{array}{c}
T_c \\
X_c(t) \\
Q_c(t) \\
Y_c(t)
\end{array} \right) \quad \Rightarrow \quad r_a
\]

\[
\begin{array}{c}
\Rightarrow \\
\Rightarrow
\end{array}
\]

\[
\begin{align*}
T & \quad X(t) \\
Q & \quad Y(t)
\end{align*}
\]

\[
\Rightarrow \quad r_c
\]
4 Systems as recursive structures
Composition plan

- Let $S_0, \ldots, S_{n-1}$ be $n$ black systems.

A composition plan for $S_0, \ldots, S_{n-1}$ is any set $C \subset \{0, \ldots, n-1\}^2$ of couples such that:

1. $\forall ((a, b), (c, d)) \in C^2, [(a \neq c) \land (b \neq d)] \lor [(a = c) \land (b = d)]$
2. $\forall (a, b) \in C$, the output $Y_a$ of $S_a$ and the input $X_b$ of $S_b$ have the same dataset.

$C$ is a set of links between outputs and inputs of $S_0, \ldots, S_{n-1}$ such that each input (resp. output) is linked to at most one output (resp. input).
Refinement of a black box

- Let $B \in BB(X, Y, Q, \mathbb{T})$.
- For all $i \in \{0, \ldots, n - 1\}$, let $B_i \in BB(X_i, Y_i, Q_i, \mathbb{T})$.
- Let $C$ be a composition plan for $B_0, \ldots, B_{n-1}$.

$(B_0, \ldots, B_{n-1}, C)$ is a **refinement** of $B$ iff the systemic signature of $C(B_0, \ldots, B_{n-1})$ is $(X, Y, Q, \mathbb{T})$. 
A **view** is a couple \((B, (B_0, \ldots, B_{n-1}, C))\):

- \(B\) is a black box
- \((B_0, \ldots, B_{n-1}, C)\) is a refinement of \(B\).
A **view** is a couple \((B, (B_0, \ldots, B_{n-1}, C))\):

- \(B\) is a black box
- \((B_0, \ldots, B_{n-1}, C)\) is a refinement of \(B\).
Concretization of a black box by a view

- Let $B_a$ be a black box.
- Let $\alpha$ be an abstraction mechanism.
- Let $V = (B_c, \_)$ be a view.

$V$ is a **concretization** of $B_a$ via $\alpha$ iff $B_c$ concretizes $B_a$ via $\alpha$. 
Realization of a view

- Let $V = (B, (B_0, \ldots, B_{n-1}, C))$ be a view.

A realization of $V$ is any realization $S_0, \ldots, S_{n-1}$ of $B_0, \ldots, B_{n-1}$ such that $C(S_0, \ldots, S_{n-1})$ is a realization of $B$. 
Realization of a view

• Let $V = (B, (B_0, \ldots, B_{n-1}, C))$ be a view.

A **realization** of $V$ is any realization $S_0, \ldots, S_{n-1}$ of $B_0, \ldots, B_{n-1}$ such that $C(S_0, \ldots, S_{n-1})$ is a realization of $B$.

In this case, $C(S_0, \ldots, S_{n-1})$ is called the **composition** of $S_0, \ldots, S_{n-1}$ **according to** $V$, and $V$ is said to be **realizable**.
A **multiscale view** $W$ is a tree such that:

- every node of $W$ is labeled with a view
- every edge $e$ of $W$ from a parent node $V_p = (\_, (B_0, \ldots, B_{n-1}, \_))$ to a child node $V_c$ is labeled with a couple $(k, \alpha)$ where:
  - $k \in \{0, \ldots, n-1\}$ is called the *index* of the edge $e$
  - $\alpha$ is an abstraction such that $V_p$ concretizes $B_k$ via $\alpha$
- for a parent node $V_p = (\_, (B_0, \ldots, B_{n-1}, \_))$, there is at most one edge of index $k \in \{0, \ldots, n-1\}$.
Multiscale view (2/2)

Boris Golden, Yann Hourdel

A minimalist formal framework for systems architecting
Free box

• Let $W$ be a multiscale view.
• Let $V$ be a view labeling a node of $W$.

A **free box** of $W$ is any black box $B$ of $V$ such that $B$ is not concretized by any child of $V$ in $W$. 
Free box

- Let $W$ be a multiscale view.
- Let $V$ be a view labeling a node of $W$.

A **free box** of $W$ is any black box $B$ of $V$ such that $B$ is not concretized by any child of $V$ in $W$.

We write $\text{freebox}(W)$ for the finite sequence of free boxes of $W$, enumerated in depth-first order.
Integration tree according to a multiscale view

- Let $W$ be a multiscale view with realizable free boxes.
- Let $S_0, \ldots, S_{n-1}$ be black systems realizing them.

**The integration tree** of $(S_0, \ldots, S_{n-1})$ according to $W$, $\mathcal{I}(W, (S_0, \ldots, S_{n-1}))$, is made by replacing the freeboxes of $W$ by $(S_0, \ldots, S_{n-1})$ and integrate everything recursively according to $W$. 

Boris Golden, Yann Hourdel

A minimalist formal framework for systems architecting
Integration tree according to a multiscale view

• Let \( W \) be a multiscale view with realizable free boxes.
• Let \( S_0, \ldots, S_{n-1} \) be black systems realizing them.

The integration tree of \( (S_0, \ldots, S_{n-1}) \) according to \( W \), \( \mathcal{I}(W, (S_0, \ldots, S_{n-1})) \), is made by replacing the freeboxes of \( W \) by \( (S_0, \ldots, S_{n-1}) \) and integrate everything recursively according to \( W \).

The integration tree is consistent iff the system labeling each node verifies the requirement of its associated black box.
Integration according to a multiscale view

- Let $W$ be a multiscale view with realizable free boxes.
- Let $S_0, \ldots, S_{n-1}$ be black systems realizing them.

The integration according to $W$ of $S_0, \ldots, S_{n-1}$ is the black system labeling the rood node of $\mathcal{I}(W, (S_0, \ldots, S_{n-1}))$. 

Integration according to a multiscale view

- Let $W$ be a multiscale view with realizable free boxes.
- Let $S_0, \ldots, S_{n-1}$ be black systems realizing them.

The integration according to $W$ of $S_0, \ldots, S_{n-1}$ is the black system labeling the rood node of $I(W, (S_0, \ldots, S_{n-1}))$.

Such an integration is consistent iff the corresponding integration tree is consistent.
Realization of a multiscale view

- Let $W$ be a multiscale view.
- Let $S_0, \ldots, S_{n-1}$ be black systems.

$(S_0, \ldots, S_{n-1})$ is a realization of $W$ if it is a consistent integration according to $W$. 
Realization of a multiscale view

Let $W$ be a multiscale view.
Let $S_0,\ldots,S_{n-1}$ be black systems.

$(S_0,\ldots,S_{n-1})$ is a realization of $W$ if it is a consistent integration according to $W$.

In this case, $W$ is said to be realizable.
A **white system** is a tree where:

- all leaves are labelled with a black system
- nodes with an even depth are labelled with a couple \((S, C)\), where \(S\) is a black system and \(C\) is a composition plan
- nodes with an odd depth are labelled with a couple \((S, \alpha)\), where \(S\) is a black system and \(\alpha\) is an abstraction function
- for each even node \((S, C)\) of children \((S_0, -), \ldots, (S_{n-1}, -)\); we have: \(S = C(S_0, \ldots, S_{n-1})\)
- for each odd node \((S, \alpha)\), its unique child \((S', -)\) is such that: \(S = \alpha(S')\).
From black boxes to white systems:

- A few heavy formalisms
- Coherent multiscale views
- Coherent systemic layers ...

... useful tools!

cf. our paper for references and complete definitions.

Boris Golden, Yann Hourdel
Conclusion

From black boxes to white systems:
Conclusion

From black boxes to white systems:
  - a few heavy formalisms

Boris Golden, Yann Hourdel
Conclusion

From black boxes to white systems:

- a few heavy formalisms
- coherent multiscale views
Conclusion

From black boxes to white systems:
- a few heavy formalisms
- coherent multiscale views
- coherent systemic layers
- ...
From black boxes to white systems:
- a few heavy formalisms
- coherent multiscale views
- coherent systemic layers
- ...

...useful tools!

Boris Golden, Yann Hourdel
A minimalist formal framework for systems architecting
Conclusion

From black boxes to white systems:

- a few heavy formalisms
- coherent multiscale views
- coherent systemic layers
- ...

... useful tools!

cf. our paper for references and complete definitions.
Thank you for your attention!

Do you have questions?