xtUML: Shlaer-Mellor Object-Oriented Analysis

Cortland Starrett
cortland.starrett@onefact.net

xtUML: Current and Next State of a Modeling Dialect

Shlaer-Mellor Object Oriented Analysis 1988-2016+

Cortland Starrett
One Fact Inc.
10412 US HWY 52 S
Clarks Hill, Indiana 47930 USA
cortland.starrett@onefact.net

***Abstract*— xtUML and its Papyrus-xtUML (BridgePoint) tooling have advanced further in the last two years than in the ten preceding years.**

Acceleration in usage and improvements in tooling have been fueled by the open source software (OSS) ecosystems of Apache, Eclipse and Papyrus. Development teams have transformed from licensed users into user-contributors.

Executable Translatable Unified Modeling Language (xtUML) is descended from Shlaer-Mellor Object Oriented Analysis (SM-OOA) and has survived the Method Wars, notation unification (UML) and tooling evolution. Sparse but focused deployment in the embedded control and distributed processing worlds has produced hundreds of years of accumulated application model intellectual property (IP).

This paper summarizes a brief history, current status and future of xtUML. It highlights recent advancements, forward direction, key players, related modeling dialects, architectural migration and implications to the metamodels and compliance with Papyrus Platform, UML2, fUML and Alf.

Keywords*— *xtUML, Shlaer-Mellor, Executable UML, UML-RT, MASL

openness, transparency and elimination of exclusive ownership fosters an environment of security. There is no single entity (licensing company) that must be trusted to serve the interests of all parties. Users can add features (or pay suppliers to add features) at will. In the xtUML community, features and fixes are being delivered faster than ever before. Related executable modeling languages and technologies are sharing in these improvements.

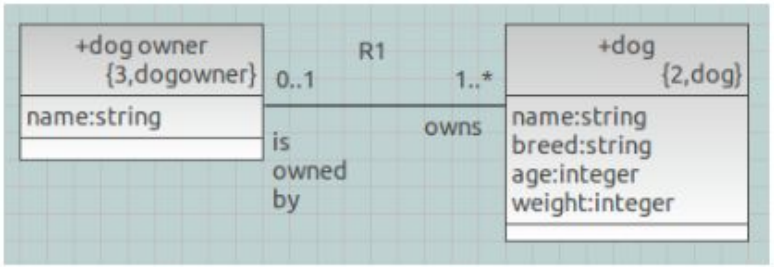
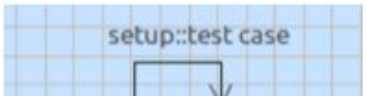


Figure 1: dog owner *owns* dog



II. Background

The Shlaer-Mellor

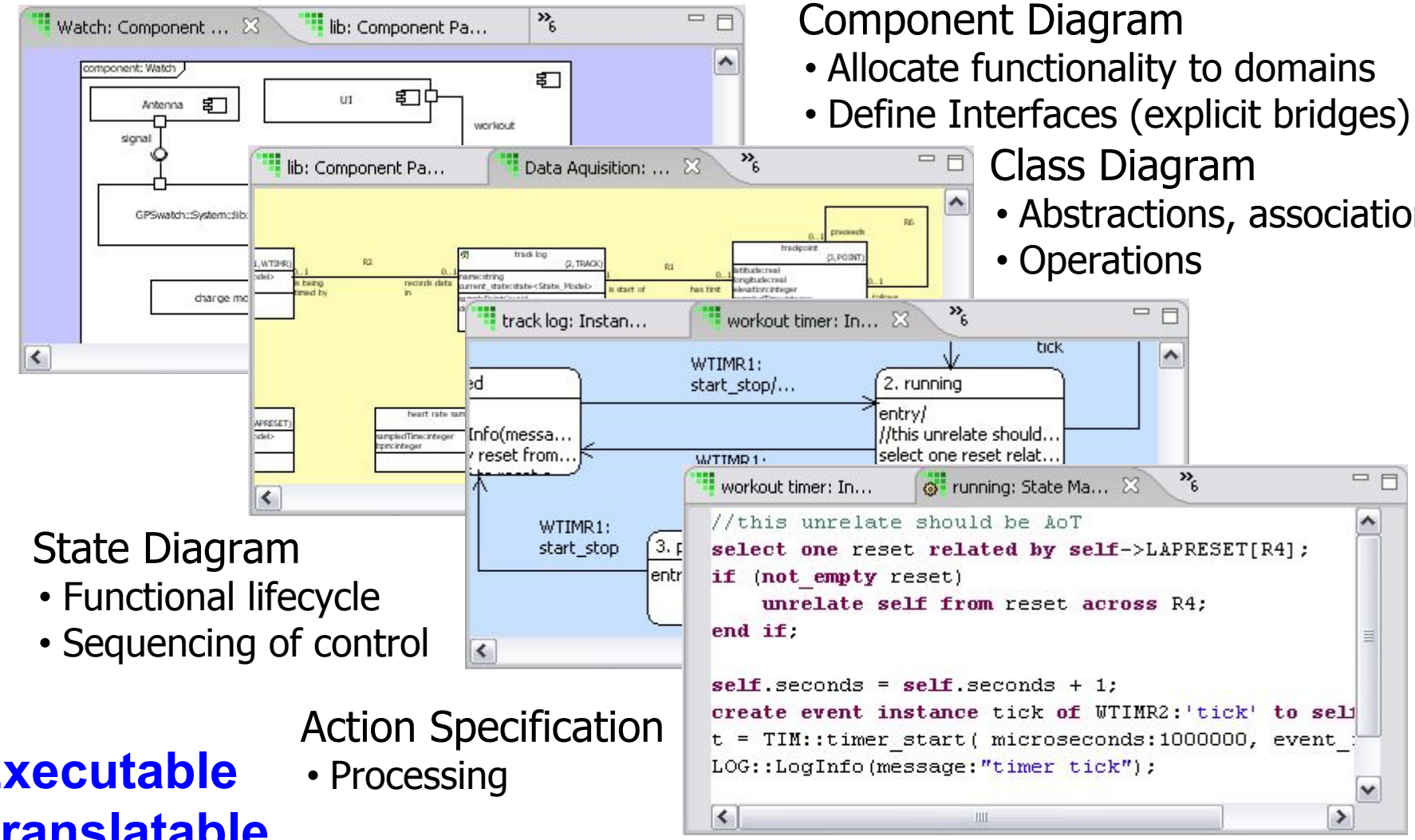
Outline

- Distinctives
- Shlaer-Mellor xtUML
- Brief History
- Key Tooling Developers
- Key Players
- Conclusion

Distinctives

- Shlaer-Mellor Method and xtUML
  - subject matters, separation of concerns
  - data, control, processing
  - tooling
- data modeling (Object Oriented Analysis (OOA))
- state machines
- action language
- interpretive execution
- model compilation

Shlaer-Mellor xtUML



Component Diagram

- Allocate functionality to domains
- Define Interfaces (explicit bridges)

Class Diagram

- Abstractions, associations
- Operations

State Diagram

- Functional lifecycle
- Sequencing of control

Action Specification

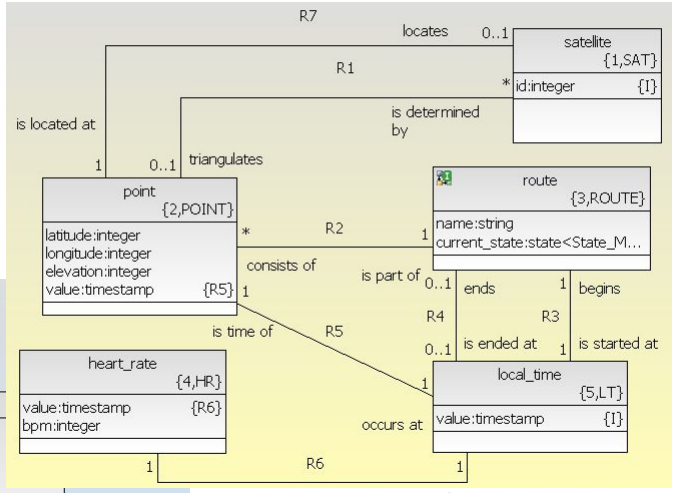
- Processing

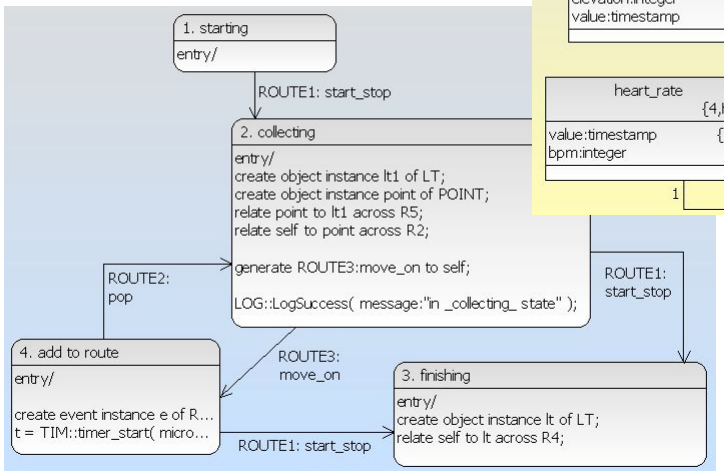
Executable
Translatable

cortland.starrett@onefact.net

2 Dimensions versus 1 Dimension

Diagrams expose abstractions…





vs.

```
 *   !!! This is an AUTO-GENERATED file.  Please do not edit.  !!!
 *-------------------------------------------------------------------------*/

#ifndef GPS_POINT_CLASS_H
#define GPS_POINT_CLASS_H

#ifdef __cplusplus
extern "C" {
#endif

#include "sys_mechs.h"
#include "gps_classes.h"
#include "sys_user_co.h"

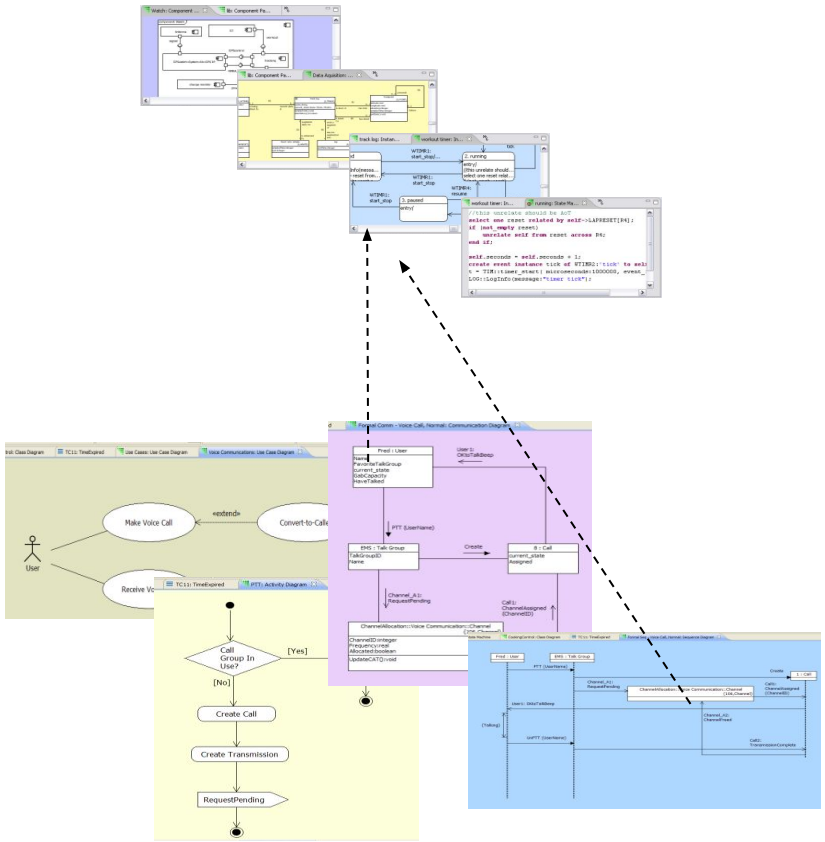
/*
 * Structure:  gps_POINT_s
 * Structural representation of application analysis class:
 *   'point'  (POINT)
 */
struct gps_POINT_s
{

  /* application analysis class attributes */
  i_t m_latitude;  /* - latitude */
  i_t m_longitude;  /* - longitude */
  i_t m_elevation;  /* - elevation */
  /* - value (R5) */  /* OPTIMIZED OUT */
```

…allowing us to reason about their correctness

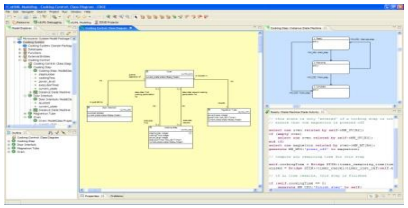
Capturing Intent in xtUML

- Executable, tightly coupled:
  - Component
  - Interface
  - Class
  - State
  - Action
- Informal, loosely coupled
  - Use Case
  - Sequence
  - Communication
  - Activity

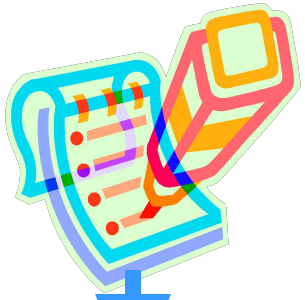


Translate Models to Target Code

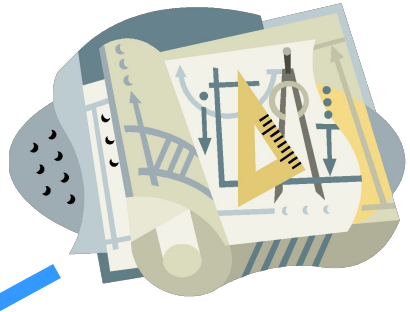
Application Models



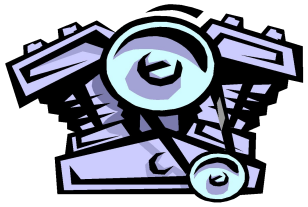
Marking Data



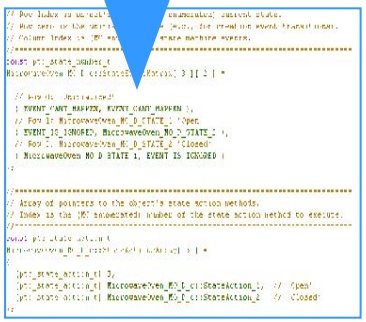
Rules & Templates



Translation Engine



Optimized Code

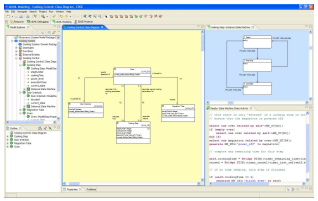


C
C++
Java
SystemC
VHDL
AUTOSAR
...

Separation of Concerns

- Subject-matter experts focus on application
  - Features and capabilities
  - Intricacies of the application

Application Models



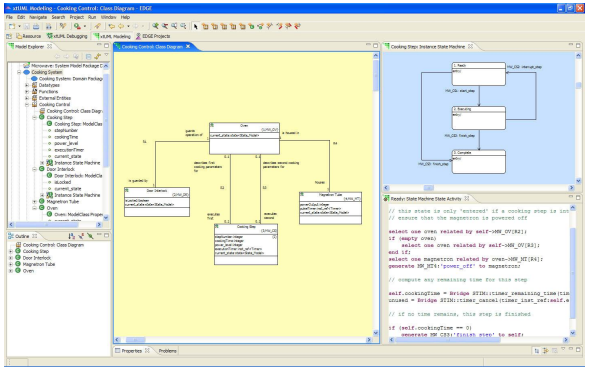
- Implementation experts focus on optimization
  - Faster, smaller
  - Less power
  - Lower cost

Model Compiler



Reusable IP (1): Models

Platform-independent
Application Models



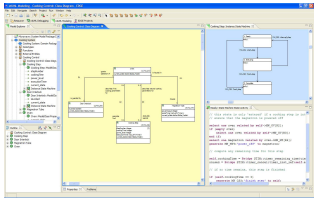


Reuse application models
across platforms and
device variants

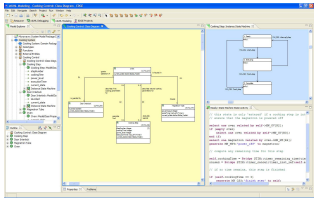


Reusable IP (2): Model Compilers

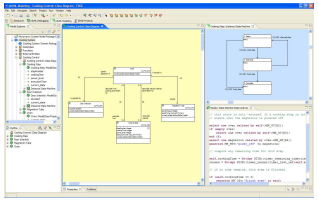
Site Link

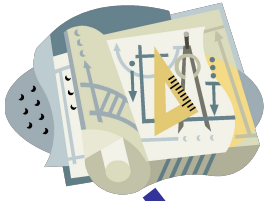


Channel Controller



MME



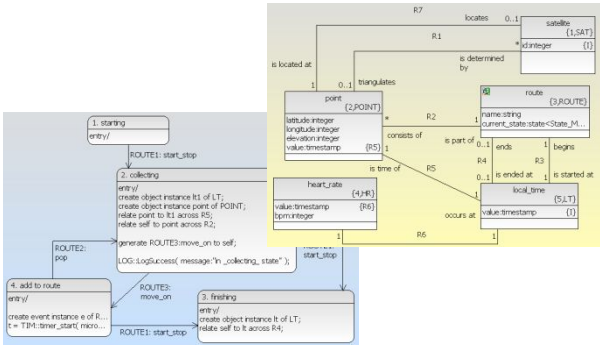


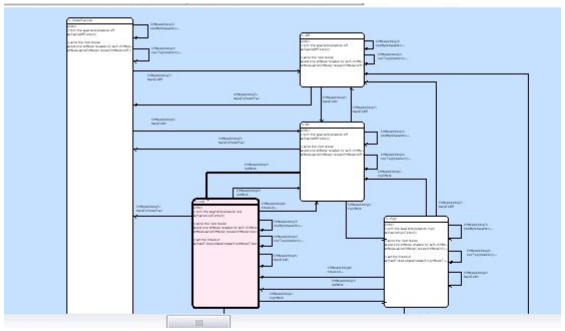
Application-independent Model Compiler

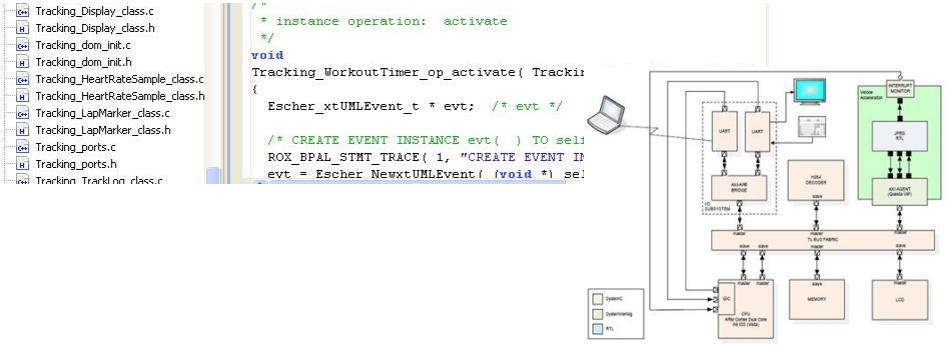
Reuse one model compiler across many applications.



Summary of the Flow







- Model
  - Create models that manifest specifications, clarify goals, and define interfaces
- Execute
  - Explore issues
  - Increase understanding
  - Prove concepts
- Translate
  - Directly generate implementations
  - Conform to standards
  - Automate quality
- Target
  - Embedded code
  - Virtual platforms
  - Deployment

Where xtUML is Used -

Embedded Systems Development











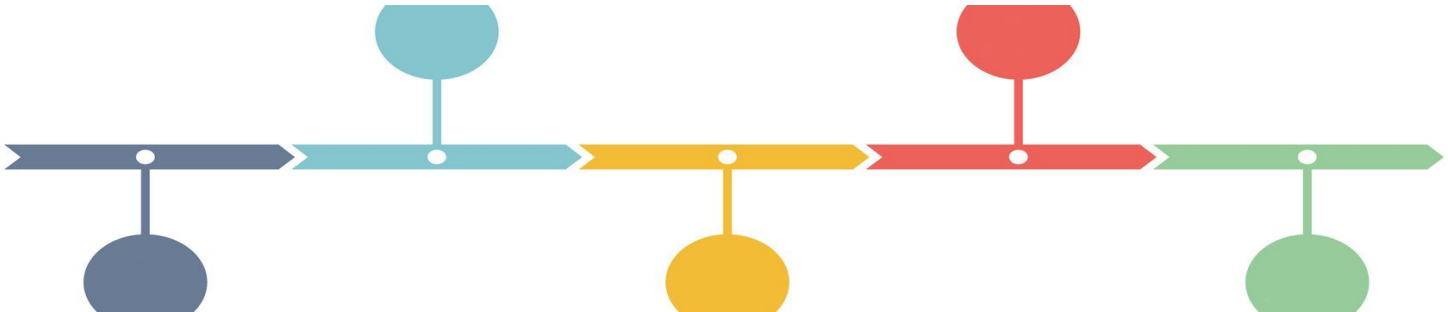




- Domain-specific
- Highly heterogeneous
- Distributed over networks
- Highly interactive w/physical world
- Require multiple disciplines to implement
- Validated mainly through physical prototyping
- System integration and test organizations are pivotal
- Subject to rigorous quality, certification, qualification rules

History

- 1988, 1991 Shlaer-Mellor Method published by Stephen Mellor and Sally Shlaer.
- 1995 BridgePoint and iOOA (iUML) tooling (commercially licensed)
- 2002 Executable UML established as Shlaer-Mellor OOA using UML notation.
- 2004 Commercial Corporate Proprietary Licensed.
- 2013 BridgePoint xtUML Editor goes open source under Apache 2.0.
- 2014 all of BridgePoint goes OSS under Apache and Creative Commons.
- 2015 Papyrus Industry Consortium and xtUML/BridgePoint contribution
- 2015 OSS of alternate generator engine (community building)
- 2016 Papyrus-xtUML (BridgePoint) Eclipse Foundation governance
- 2016 OSS contributions from industry, university and individuals



Key Users

- Saab
- UK Crown
- Agilent
- Ericsson
- Fuji-Xerox
- BAE
- …
- Academia



SAAB

VRICON

BAE SYSTEMS





Agilent Technologies



Canon



ERICSSON

SONY

DELPHI
Automotive Systems



九州大学
KYUSHU UNIVERSITY



SHINSHU
UNIVERSITY

FUJI Xerox





CHALMERS
UNIVERSITY OF TECHNOLOGY

faith
CHRISTIAN SCHOOL

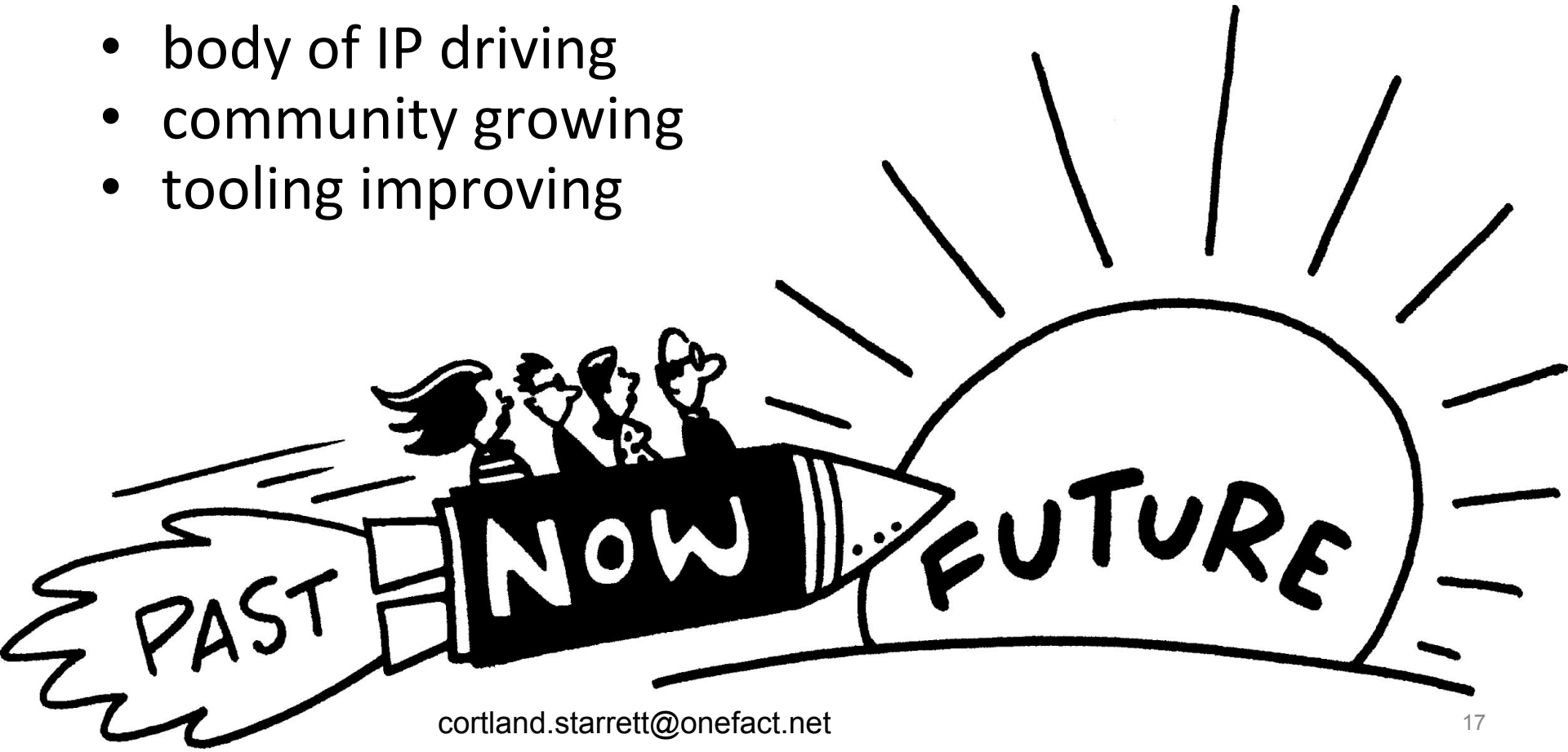
LINKÖPING
UNIVERSITY

Key Development

- Project Technology
- Cadre (Teamwork)
- Kennedy Carter
- Mentor Graphics
- Abstract Solutions
- Toyo
- Aurora Consulting
- Model Integration
- ROX Software
- One Fact

Conclusion

- method sound
- body of IP driving
- community growing
- tooling improving



Questions and Discussion



Methodology

UML-RT (ROOM)	xtUML (Shlaer-Mellor)
elaborative	translative
functional decomposition	domain based
encapsulation	relational
executable	executable
abstract	more abstract
architecture recognized	architecture domain (MC)
fixed run-time	run-time rules

Workflow

UML-RT

1) top capsule
2) decomposition
3) messaging/protocol
4) state behavior
5) progressive refinement
6) activities

xtUML

1) domain separation
2) classes, associations and attributes
3) state behavior
4) activities
5) component deployment

Tool Architecture

