TECHNISCHE UNIVERSITÄT DRESDEN

Department of Computer Science Institute for System Architecture, Chair for Computer Networks

Computer Networks Group

Engineering Self-Adaptive Systems

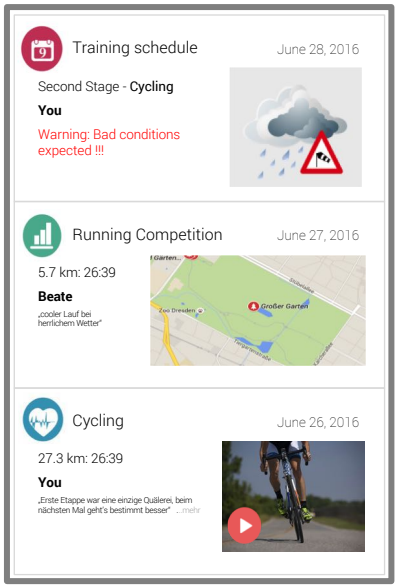
Profilmodul Forschungslinie

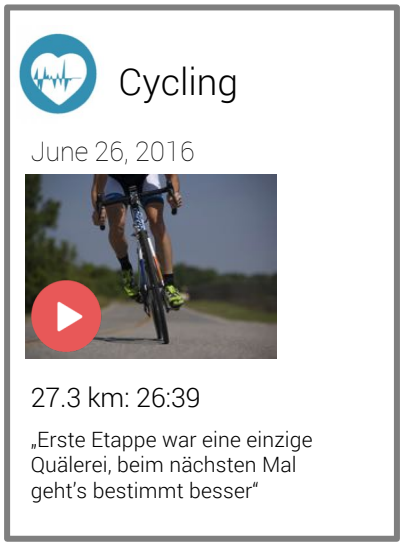
Prof. Dr. Alexander Schill
Dr. Thomas Springer

Motivating Example

- Social Fitness App
  - Wristband/Smart Watch + Smartphone App
  - Server component for data storage and user management

- Functionality
  - Automatic detection and tracking of sports activities
  - Recording of activity states and content (images, videos, track record, curve with pulse, etc.)
  - Activity Timeline - Posting own activities and see others‘ activities in an integrated timeline with text, images, videos, etc.
  - Management of training schedule and planning of training activities
  - Managing competitions with ranking





Problem Analysis

- Connectivity
  - Low data rate and high delay in cellular networks
  - Intermittent connectivity due to handover and network changes
- Offline Phases
  - Limited coverage, unavailability of servers
  - To save energy or communication cost
- Form Factor
  - Limited Resources (CPU, memory, display size)
  - Diversity of device classes (Smart Watches, Smartphones, Tablets) and device platforms (iOS, Android, Windows Mobile)
- Energy
  - Limited battery capacity
  - Display, computation, communication and sensing consume energy

Adaptation Examples

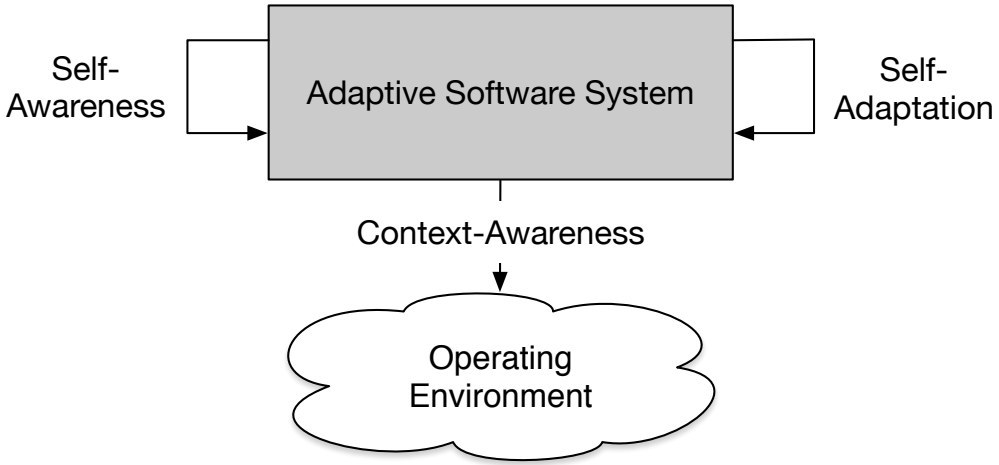
- Automatic detection and tracking of sports activities
  - Accelerometer based activity tracking
  - E.g. GPS is activated only if running or cycling activity to save energy
  - Higher sensing frequency is applied when walking or running is detected
- Sharing (upload and download) of activities to social network
  - Queueing for upload and download
  - Filtering (cycling related activities only), lossy conversion and lazy evaluation for data download
  - Prefetching and Caching of data (e.g. proactive download of activity content)
- Scheduling of training activities
  - Weather conditions monitored -> rescheduling outdoor activities if it is raining
- Network connection loss
  - Using of cached data for timeline, local storage of tracked data

What is a self-adaptive system?

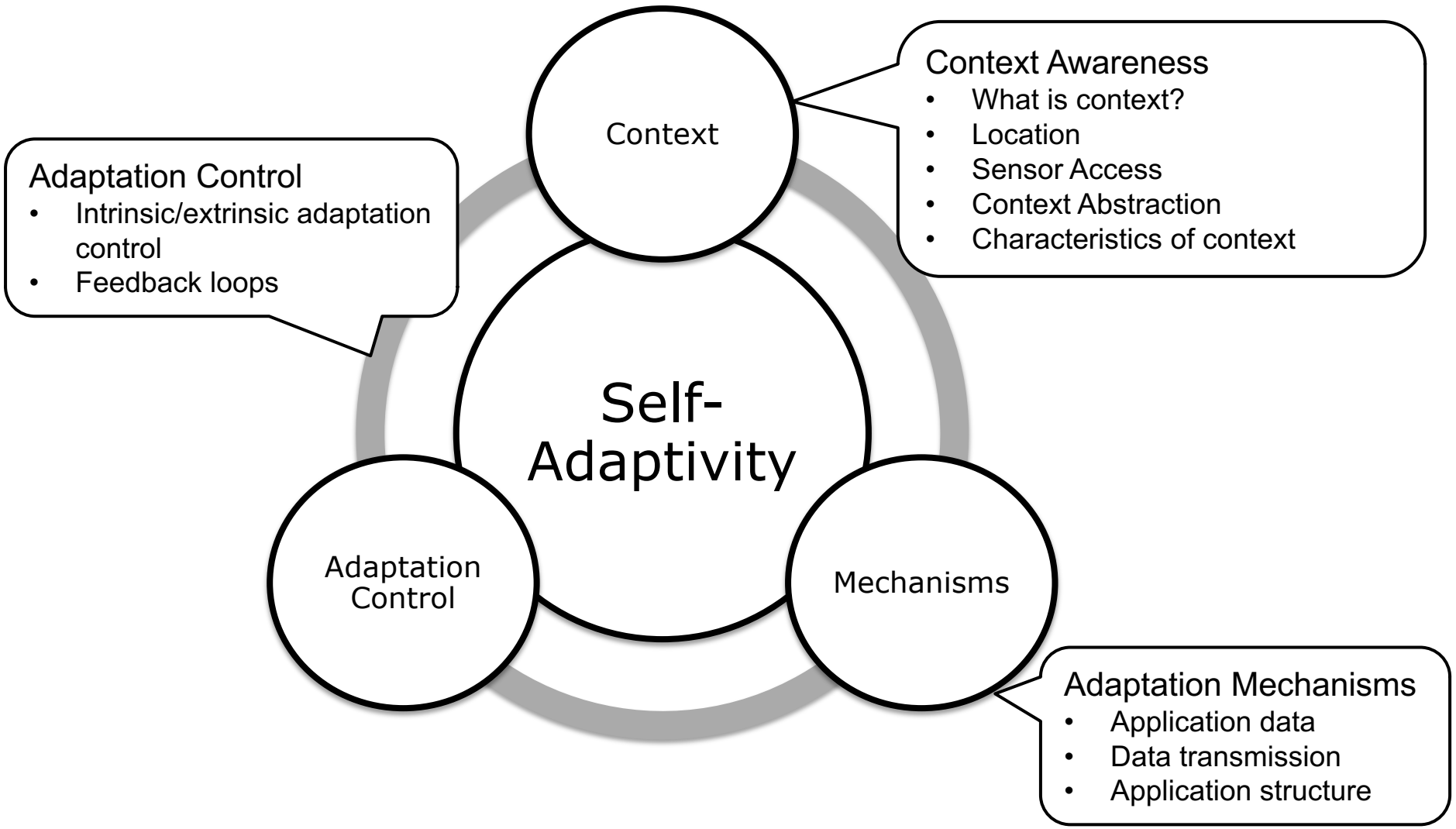
„Self-adaptive software modifies its own behavior in response to changes in its operating environment. By operating environment, we mean anything observable by the software system, such as end-user input, external hardware devices and sensors, or program instrumentation.“ [Oreizy et.al]

Three mandatory requirements:

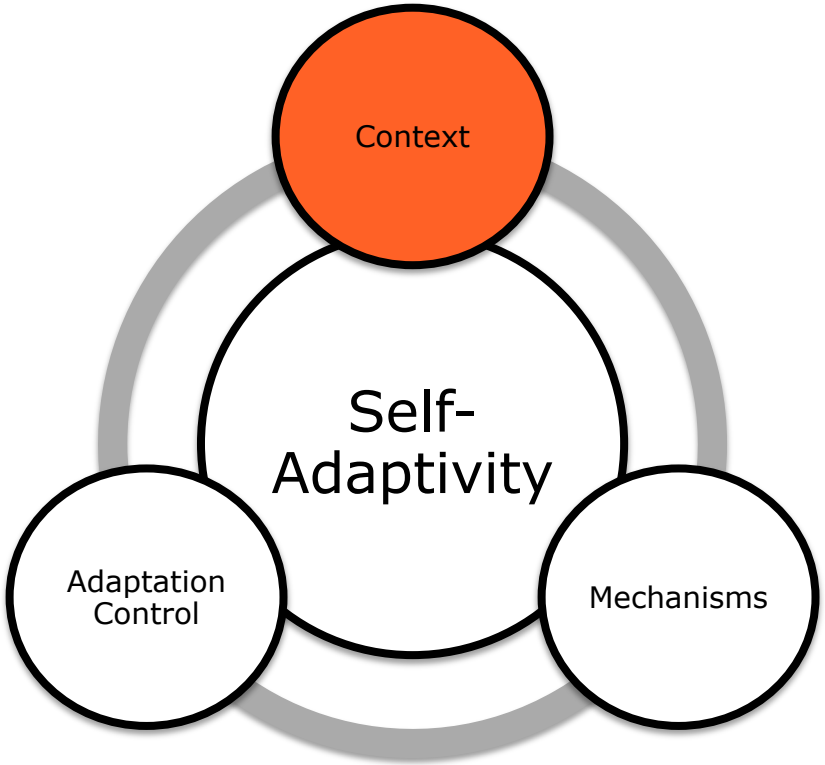
- Context-Awareness:
  - aware of operational environment
- Self-Awareness:
  - knowledge about current configuration or state
- Self-Adaptation:
  - modify itself at runtime



Lecture Overview



CONTEXT



Definition of Context

- *"… all but the explicit input and output of an application"* [LiS00]
- *"… that which surrounds, and gives meaning to, something else"* [o.V.00]

- "*Context is a subjective concept that is defined by the entity that perceives it" and „contextual states [...] are inherently associated with specific objects"* [Pascoe]

- *„Context is an operational term: Something is context because of the way it is used in interpretation, not due to its inherent properties. [...] Features of the world become context through their use."* [Winograd]

- "*Context is any information that can be used to characterise the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves"* [Anind K. Dey]

Types of Contextual Information

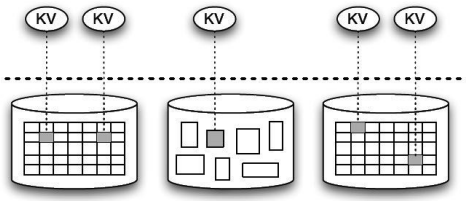
Contextual dimension	Respective contextual information
Physical context	Location, time, temperature, light and noise intensity, nearby persons
Technical context	Network (bandwidth, latency, error rate), Device (input and output capabilities, memory, software support), available services, service preferences
Personal context	Address, phone number, payment, preferences, schedule, service preferences
Social context	Nearby persons, groups (teams) to which the user belongs
Operational Context	Roles, activities, to-do-items, content of the inbox of the user

Types Relevant for Example App

Contextual dimension	Respective contextual information
Physical context	Location, time, temperature, light and noise intensity
Technical context	Network (bandwidth, latency, error rate), Device (input and output capabilities, memory, software support), available services, service preferences
Personal context	Address, phone number, payment, preferences, schedule, service preferences
Social context	Nearby persons, groups (teams) to which the user belongs
Operational Context	Roles, activities, to-do-items, content of the inbox of the user

Possible Context Sources

- Context Sources are:
  - sensor devices (e.g. GPS, temperature, light intensity, noise)
  - databases (e.g. extracting context from structured data in DB)
  - application data (e.g. scheduling app)
  - user monitoring and input (e.g. location or task from current user activity)



Context Capturing
Sensor Access in Android

- Access via the Android SDK
  - **android.hardware.SensorManager**
  - `sensorManager = (SensorManager) getSystemService(SENSOR_SERVICE);`
- Sensor supported?
  - `sensorManager.getSensorList(SensorManager.SENSOR_ALL)`

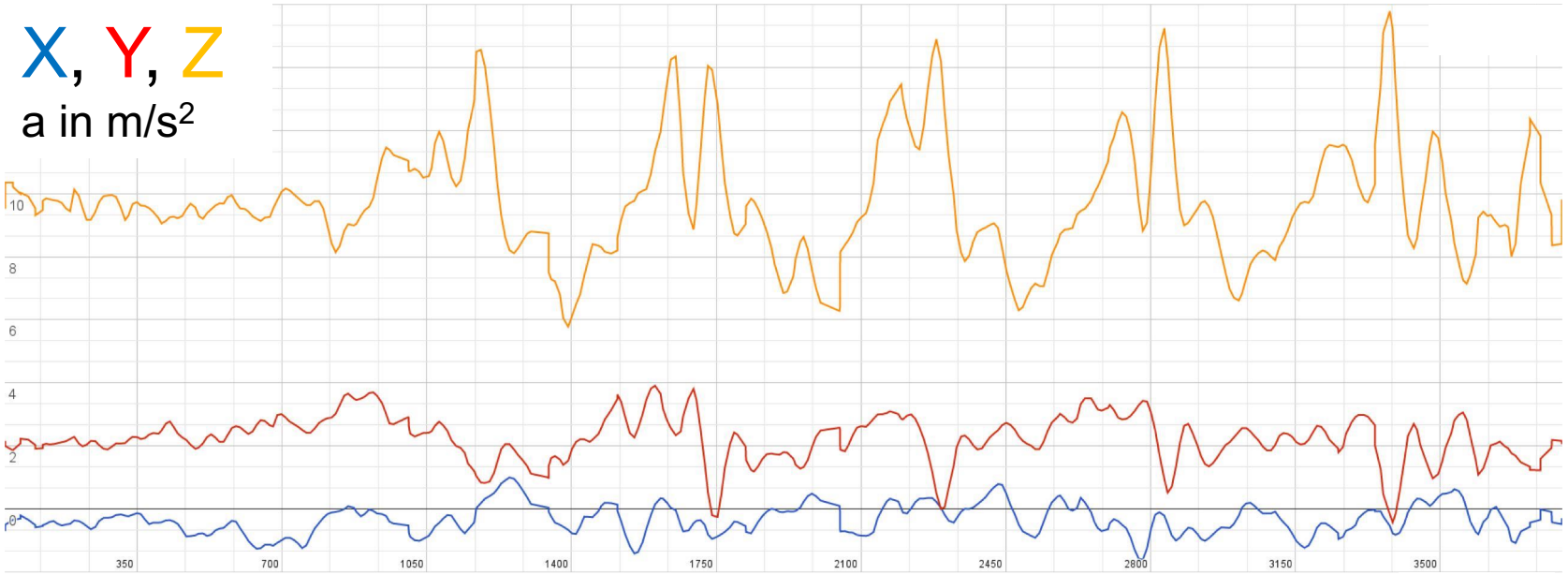
- EventListener observes changes in sensor data

```
class AccListener implements SensorEventListener {
    public void onAccuracyChanged(Sensor sensor, int accuracy) {…}
    public void onSensorChanged(SensorEvent event) {…}
```

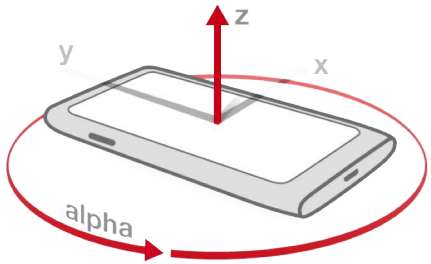
- Register for the updates when Activity is in foreground

```
protected void onResume() {
 super.onResume();
 sensorManager.registerListener(accListener,
     sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER),
     SensorManager.SENSOR_DELAY_NORMAL);
}
```

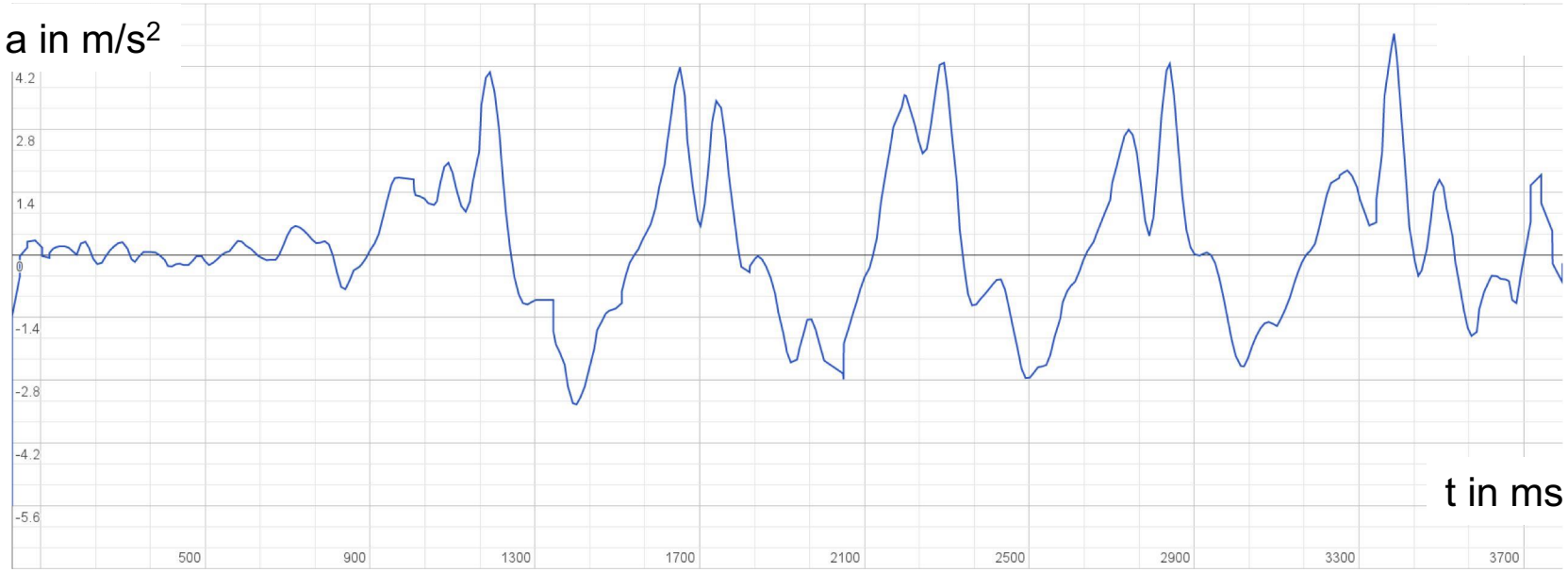
Abstracting low level context information



- accelerometer data record of a walk
- each axis is represented as a line
- significant pattern for different types of movement
  - e.g. pattern for steps on z-axis
- problems to consider:
  - result is axis dependent
  - gravity has impact on accelerometer

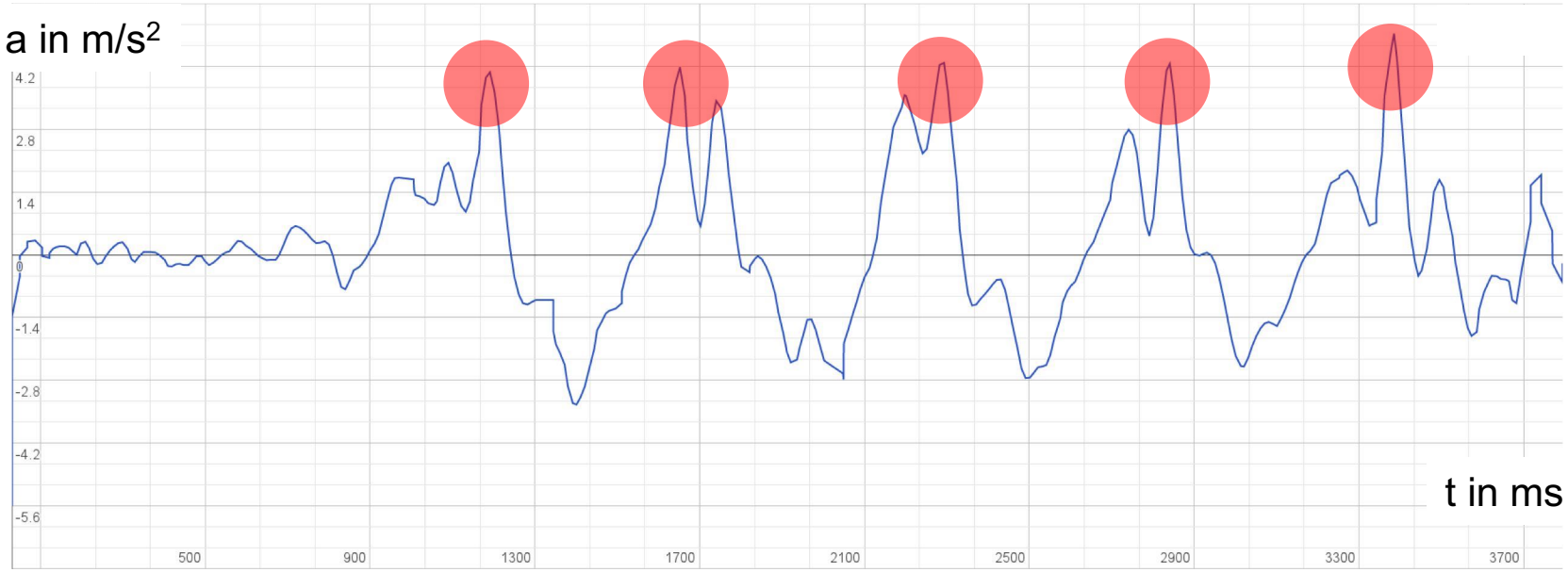


Step Detection via Accelerometer



- Processing raw sensor data
  - root mean square over all 3 axes
  - removal of gravity
  - low pass filtering of the raw data
- Result: significant, orientation independent curve

Recognizing the activity

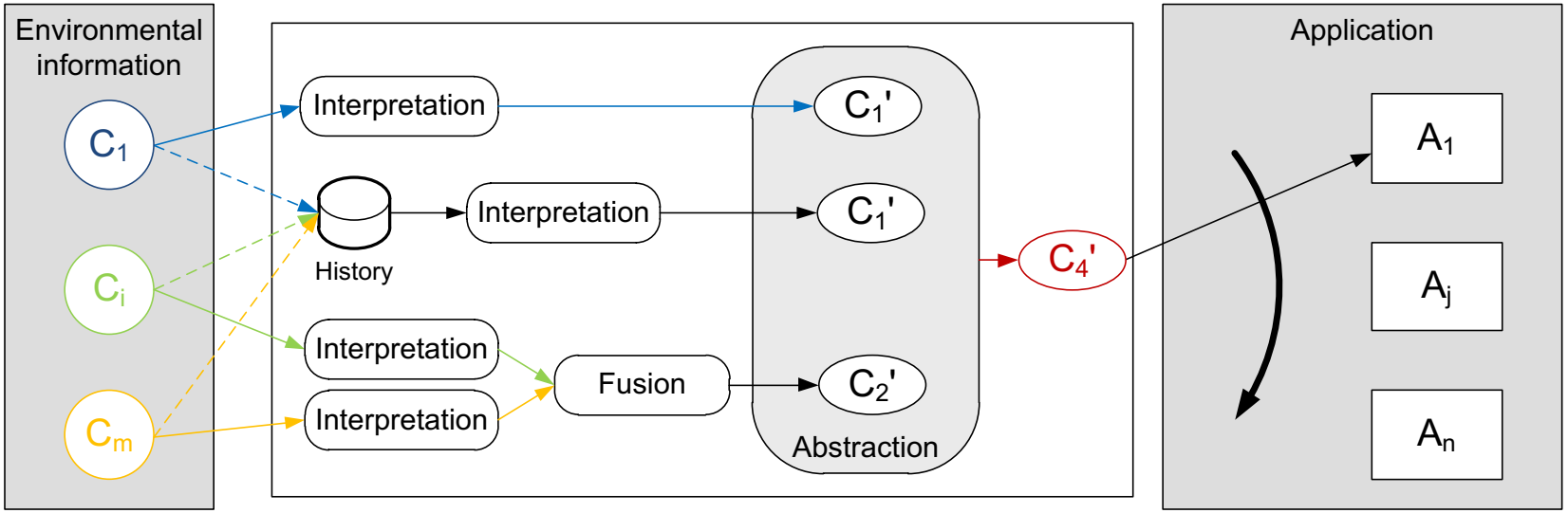


- Detecting steps based on peaks
- Recognizing the activity based on:
  - No. of steps
  - with step length, distance can be derived
  - Step frequency
  - Speed

➢ Activity = Running if speed and step frequency are within certain tresholds

Context Aggregation

temperature (12°) cold cloud formation, recommend user
humidity (90%) high humidity high probably of to reschedule
air pressure (1005 hPa) high air pressure rain and storm cycling activity



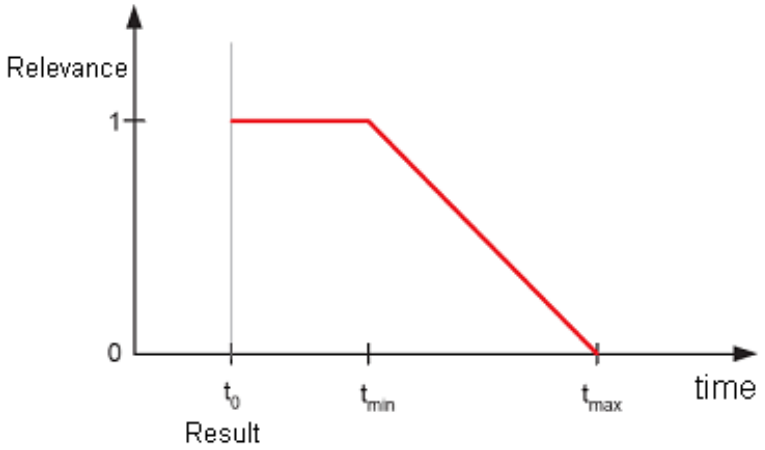
C = ContextValue

A = Action

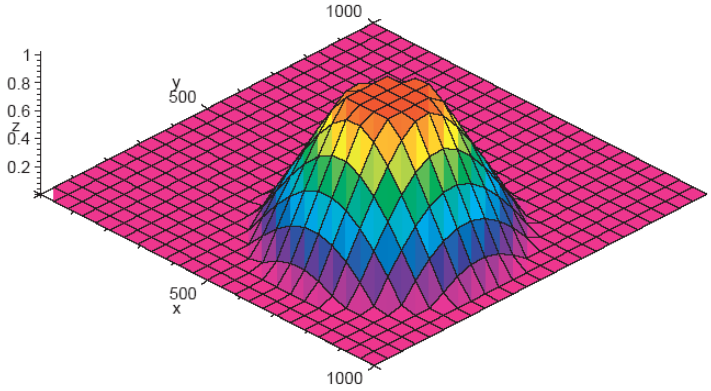
Context Capturing → Context Abstraction → Context Usage

Properties of Context Attributes

- Time dependent
  - highest relevance at capturing time -> decreases constantly
  - History: represents values at different points in time



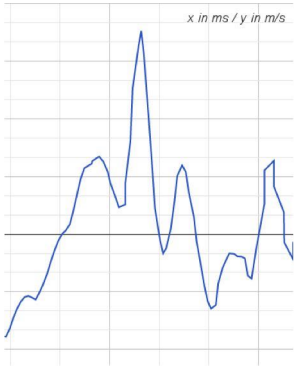
- Location dependant
  - highest relevance at capturing place -> decreases with distance

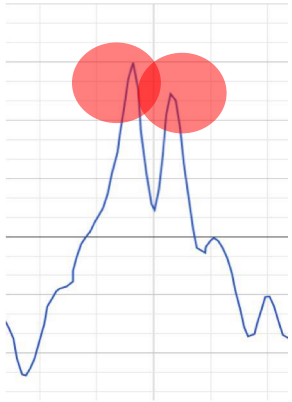


Properties of Context Attributes

Imperfection

- Unknown – sensor failure or unavailability
  - e.g., GPS does not work indoors
- Erreneous – due to measurement failures or wrong assumptions for derivation
  - e.g., no clear step pattern can be detected
  - e.g., wrong value for step length leads to wrong distance
- Inaccurate – due to uncertainty in measurements, application of heuristics
  - e.g., too many steps are counted due to wrong peek interpretation
- Ambiguous – due to conflicting alternative values
  - e.g., distance provided by acc and GPS is different





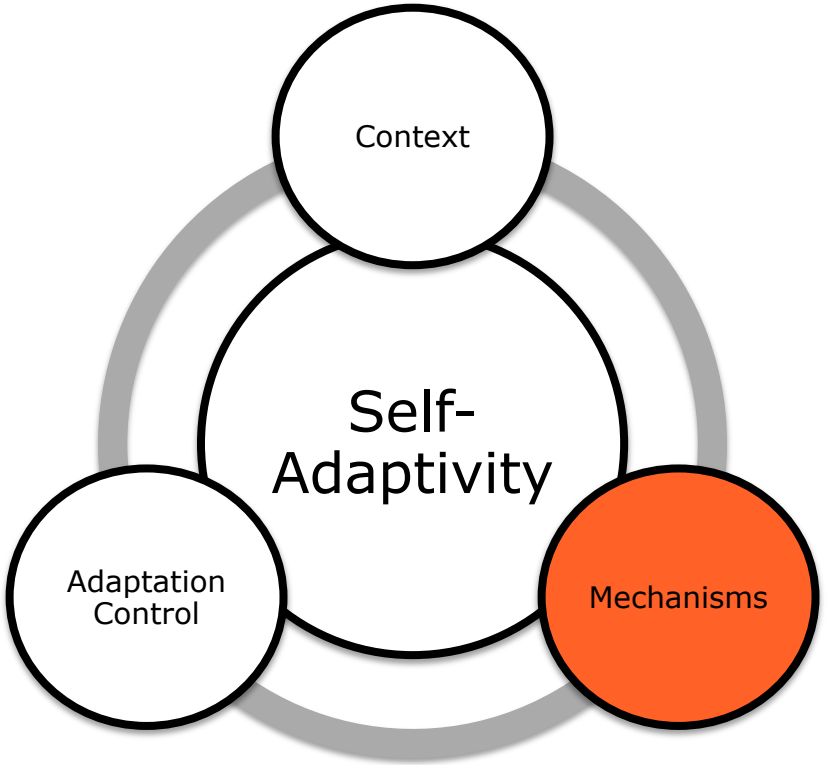
Context Summary

- Context information enables awareness of excecution environment
- Pre-requisite to control adaptation processes

- The major phases of context processing
  - Gathering
  - Abstraction
  - Decision making

- Context requires special handling since it is not explicitly available and gathering process results in special characteristics of content (e.g. unavailability, uncertainty, inconsistency)

ADAPTATION MECHANISMS



Adaptation Mechanisms

- Responsible for executing adaptations
- Taking knowledge about application into account -> self-awareness

- **Object** of Adaptation (assuming a smartphone App with a backend)
  - Application data
  - Data transmission
  - Application structure

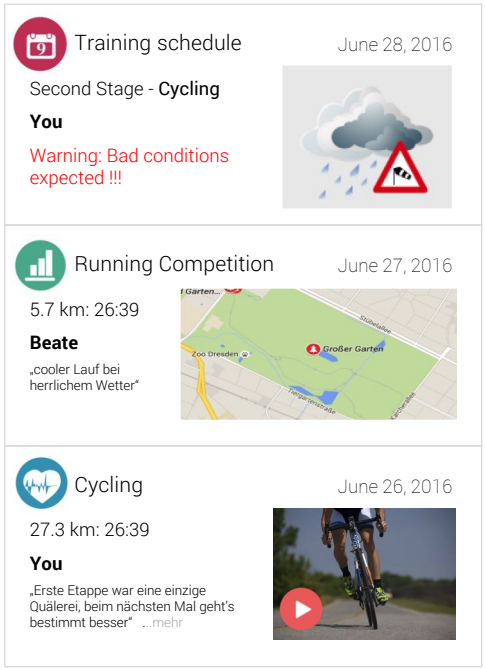
Filtering
Adaptation of Application Data

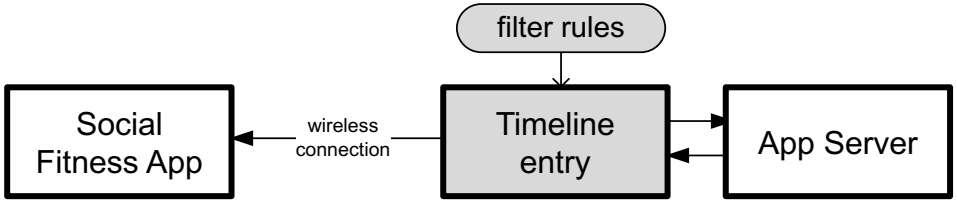
- Goal: reduction of data volume
  - Coarse grainded approach – keep or drop
- Low consumption of computation time and resources
  - no data processing but data selection and cancelation
- Filter rules for data selection defined
- Filtering possible based on different criteria
  - External criteria (dropping of packets in router according to threshold for queue length)
  - Data structure (dropping all email attachments, dropping all P and B frames of mpeg stream)
  - Data type (dropping videos in web pages)
  - Data format (dropping png images of web pages)
  - Semantics (dropping of private emails based on keywords)
- Usually performed at runtime

Filtering – Example
Adaptation of Application Data

Example: Activity Timeline

- Header:
  - Filter activities of particular type (e.g. cycling, outdoor)
  - Select activites for certain competition only (defined group of partitipants AND type of sports AND location)
- Body:
  - Filter activities by keywords in description (e.g. Team Challenge Dresden, new record)
- Attachments:
  - Discard attachments larger than a treshold (e.g. large images)
  - Discard all attachments of type video





Lossy Conversion

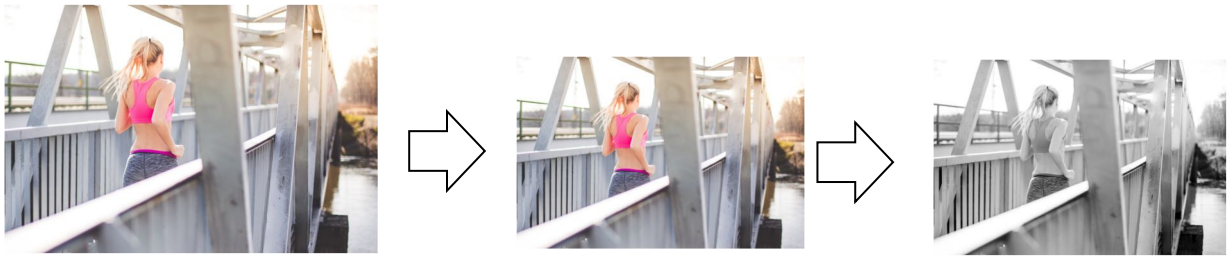
- Goal: reduce data volume but limit content loss
- Operates on source data
- Creates related data objects of **similar type**
  - Keeping the content to some degree
- Operations dependent on data/media type
  - Change of specific properties of media objects
    - Image (resolution, colour depth)
    - Video (frame rate, size, colour depth)
    - Audio (sample rate, sample size)
    - Specific types (document, business object, etc.)
- More fine-grained reduction of media data dependent on device capabilities (e.g. display size or storage capacity) and connectivity

Lossy Conversion - Example
Adaptation of Application Data

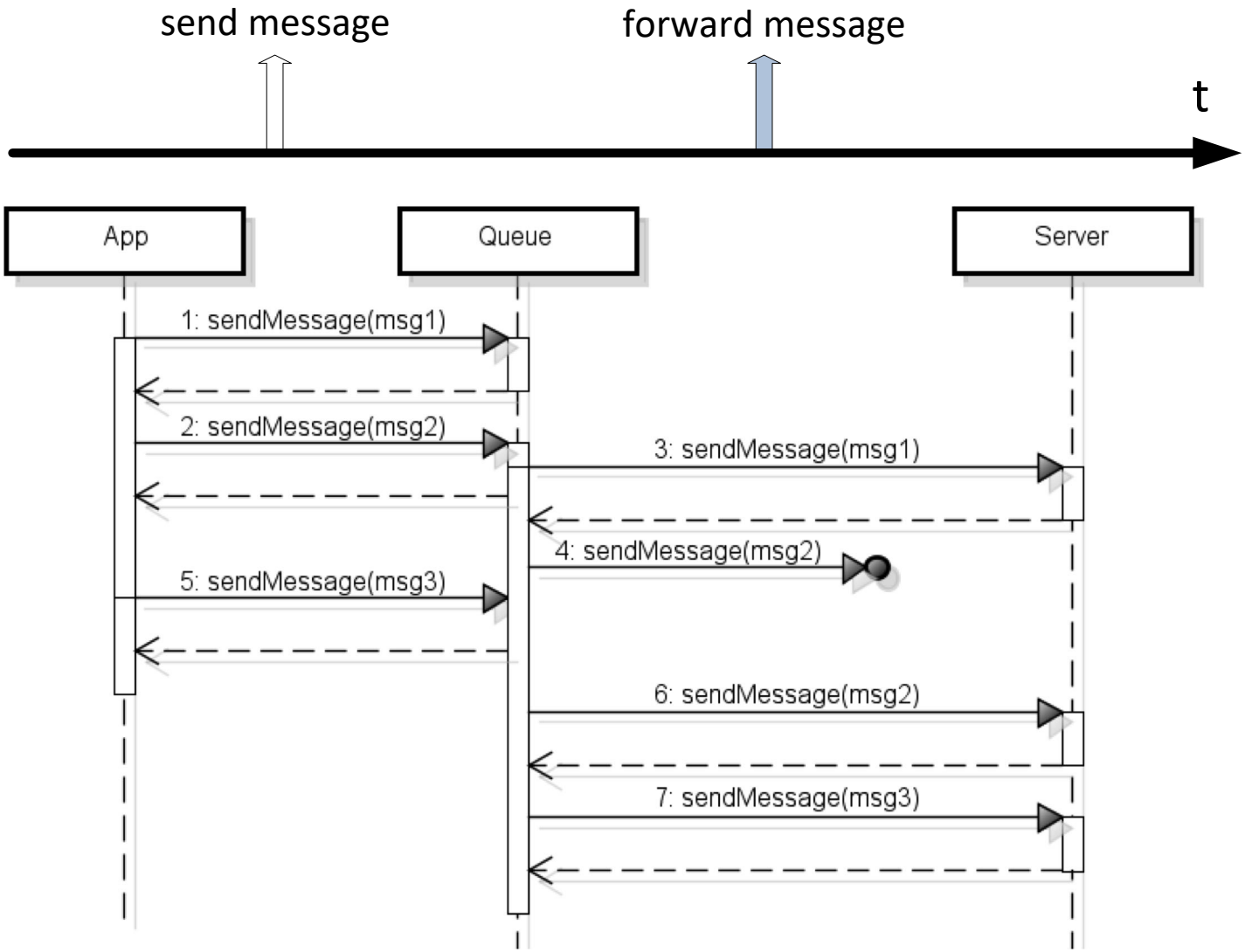
- On-the-fly adaptation at runtime
  - Use current context (bandwidth, display size, etc.) to adjust parameters of data objects

  \+ Increased flexibility of adaptation
  \+ No additional storage requirements
  – Consumes processing resources at runtime
  – Can increase response time

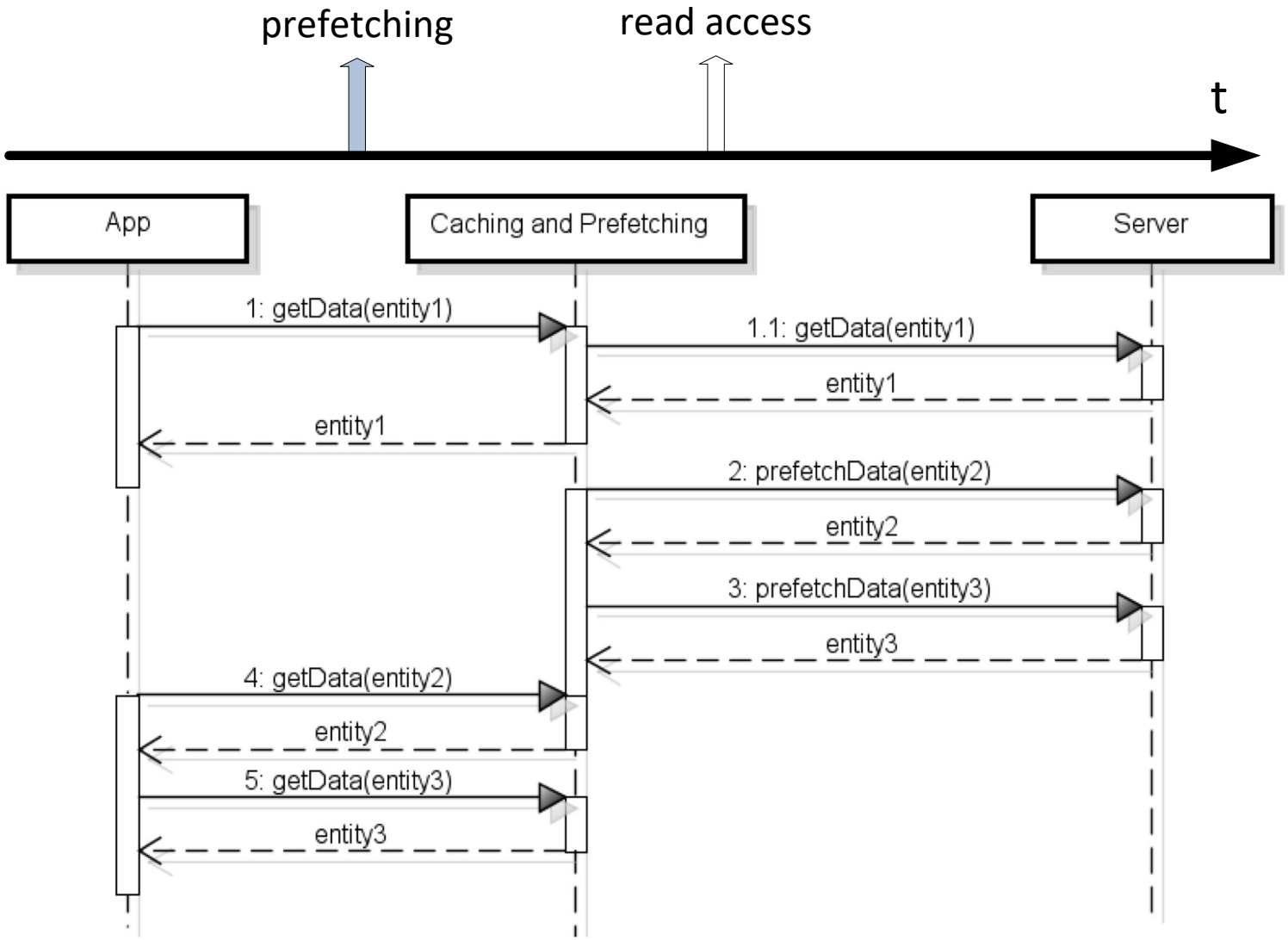
- Example: Lossy conversion of image to reduce transfer size



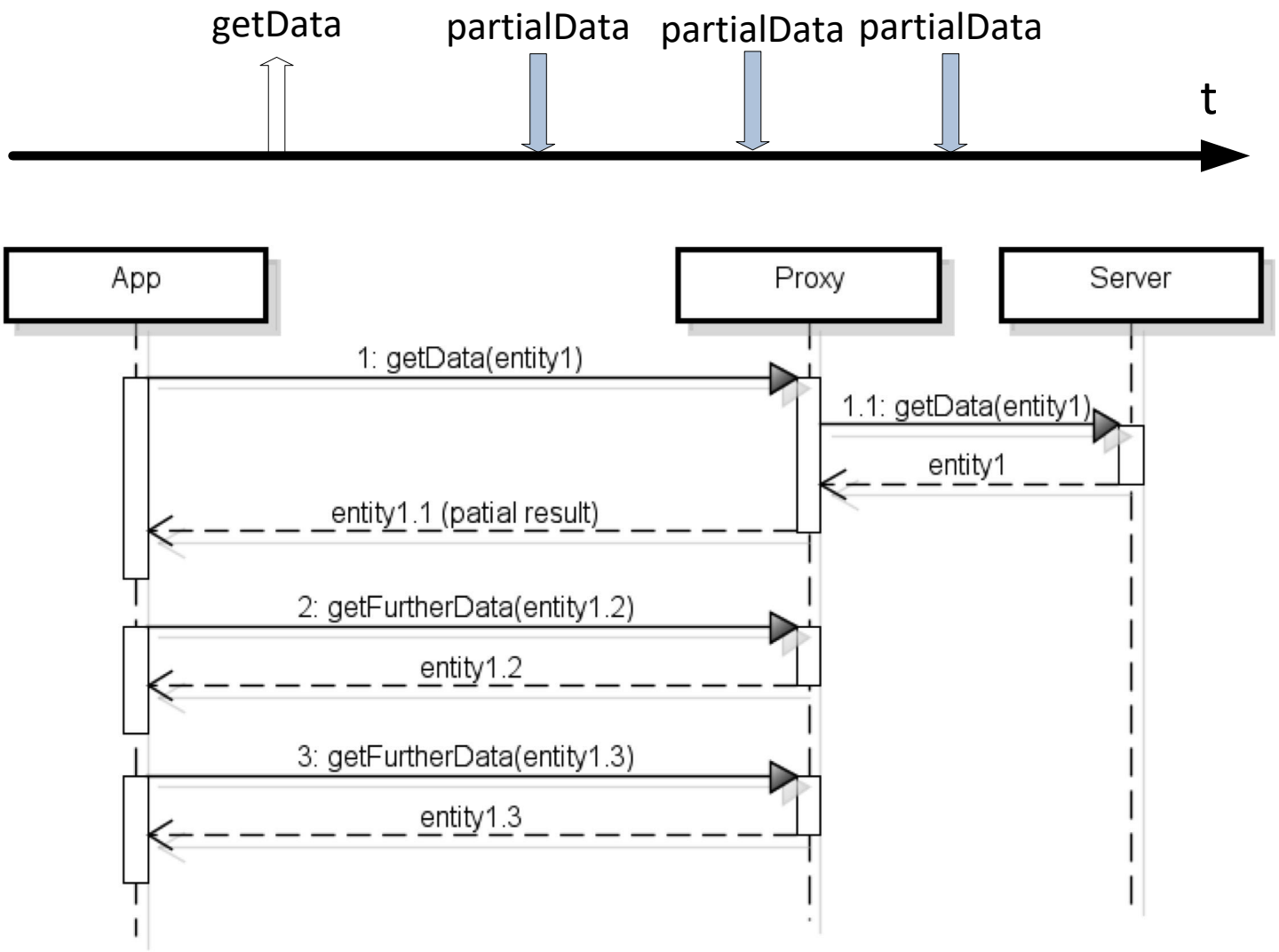
Queuing
Adaptation of Application Data



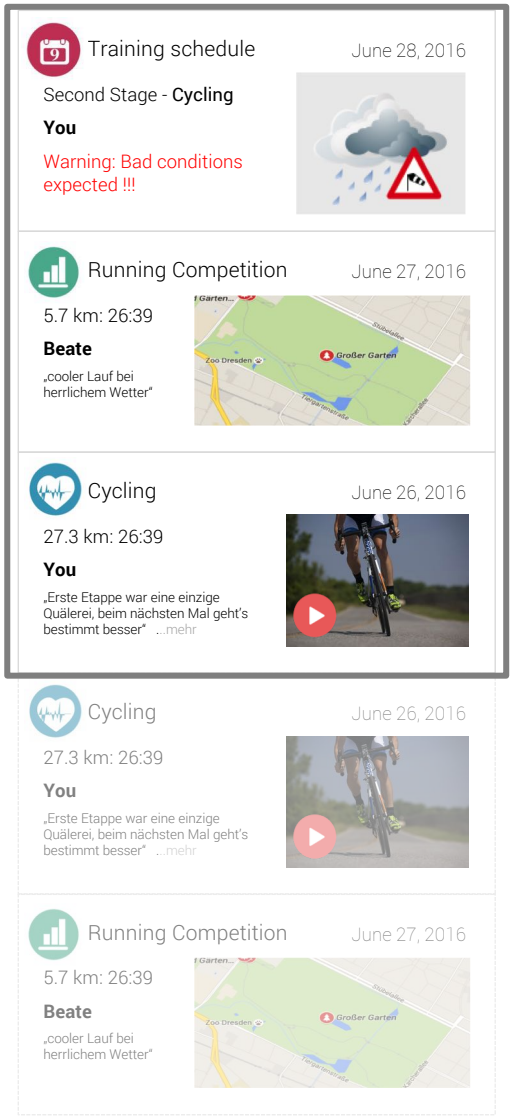
Prefetching
Adaptation of Data Transmission



Lazy Evaluation
Adaptation of Data Transmission



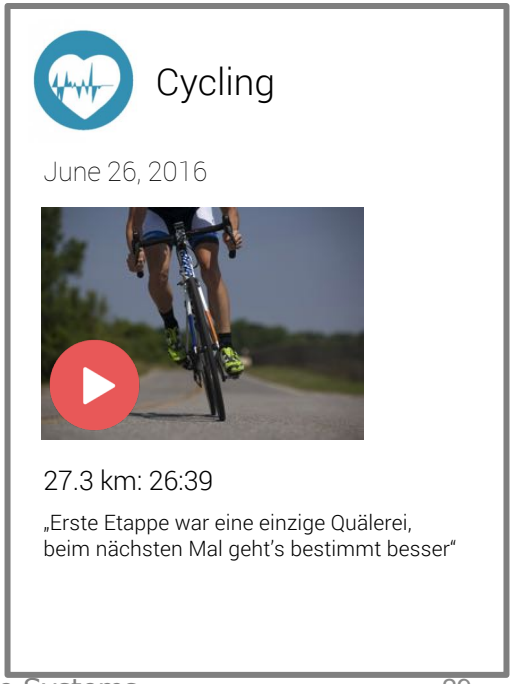
Lazy Evaluation - Example
Adaptation of Data Transmission



- Loading of the first three visible entries in the timeline only
- Only if user starts scrolling, the next entries in the timeline are loaded
- Details for entries, e.g. full video or full size images, are only loaded, if user selects detail view for timeline entry

Detail View

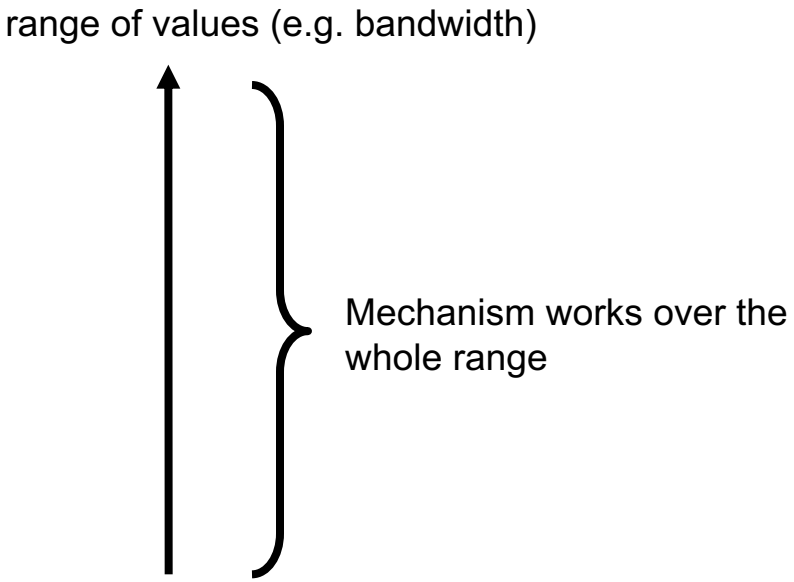
Load when scrolling



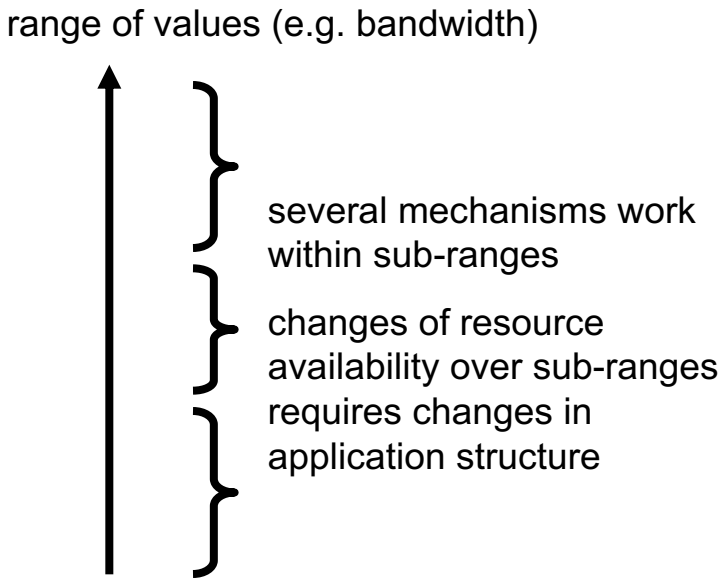
Parameter and Structure Adaptation

- changing the adaptation strategy might require changing the structure of the software system

Adaptation by Parametrization

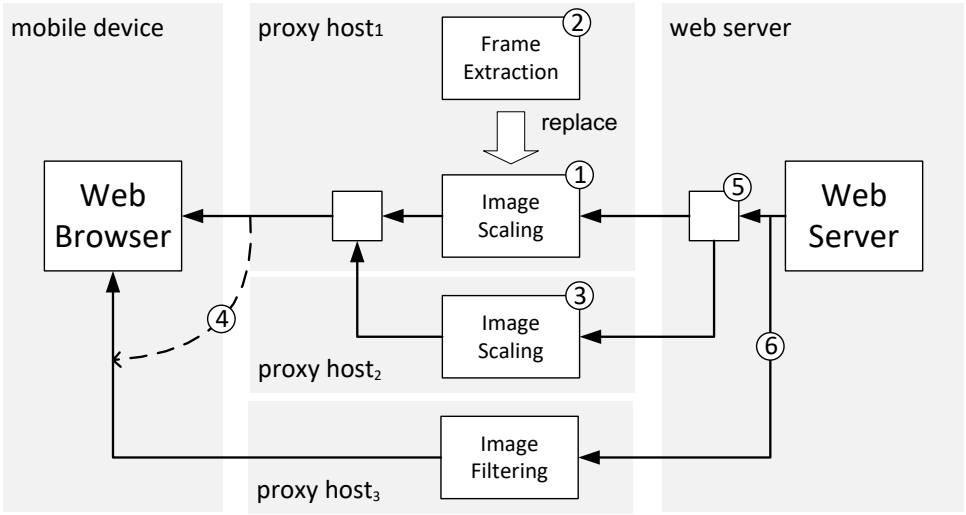


Adaption by Structural Changes



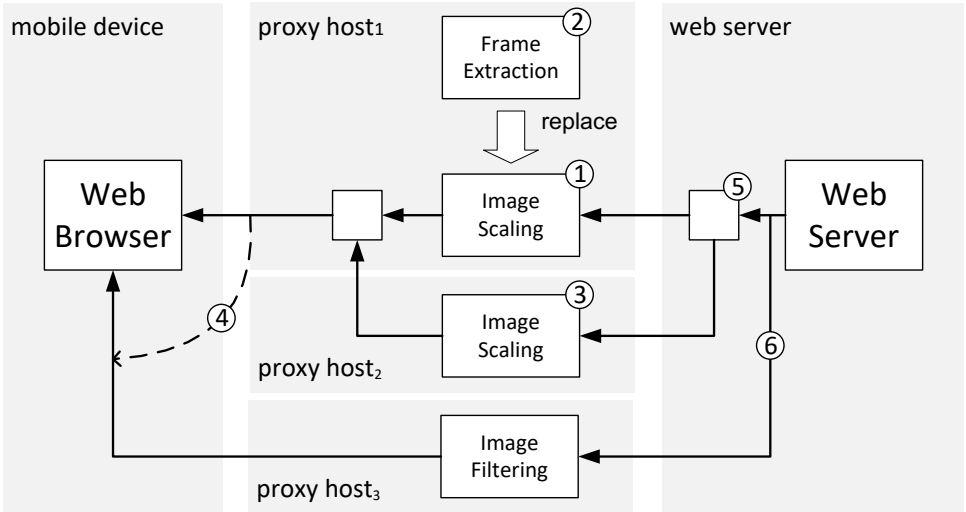
Adapting components

- Adding/Delete
  1. Add image scaling component to reduce data volume
- Replacing
  2. Replace component for image scaling with component for extracting image frames
- Replication
  3. Replicate image scaling component for load balancing



Connections

- Dynamic binding
  4. Rebind to an alternative proxy
- Branching
  5. Use alternative to balance load
- Parallelisation
  6. Use parallel paths where copies of data are processed in different ways to provide multiple versions of content
- Indirection/Proxy
  - Use proxy pattern to decouple components (e.g. Split-TCP)



Adaptation Mechanisms Summary

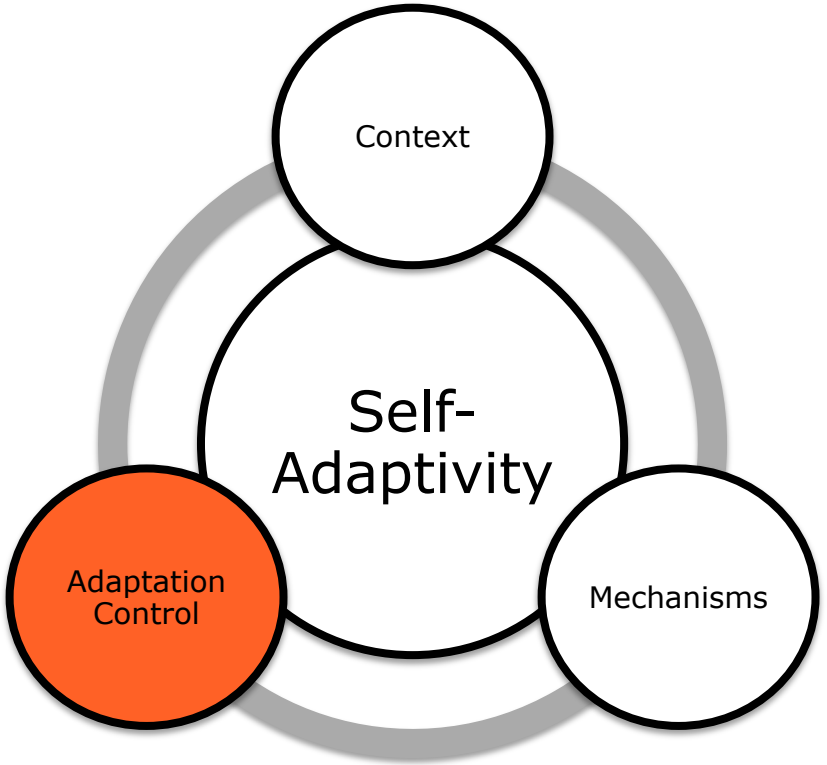
- Adaptation as a principle how to build applications
- Mechanisms are adoptable in different contexts

- Large variety of adaptation mechanisms available
  - Application data
  - Data transmission
  - Application structure

- Mechanisms can be combined

➢ Control of adaptation process necessary

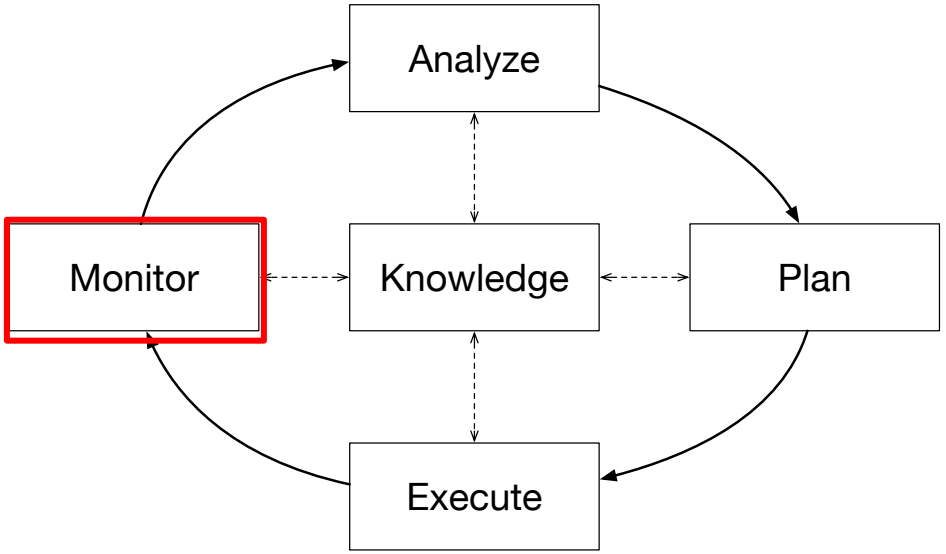
ADAPTATION CONTROL



MAPE-K
Adaptation control

- MAPE-K feedback loop as architectural reference for designing self-adaptive software systems

- M – Monitor
  - Collecting data from sensors and further data sources
  - Correlating measurements to higher level context
- A – Analyze
- P – Plan
- E – Execute
- K – Knowledge

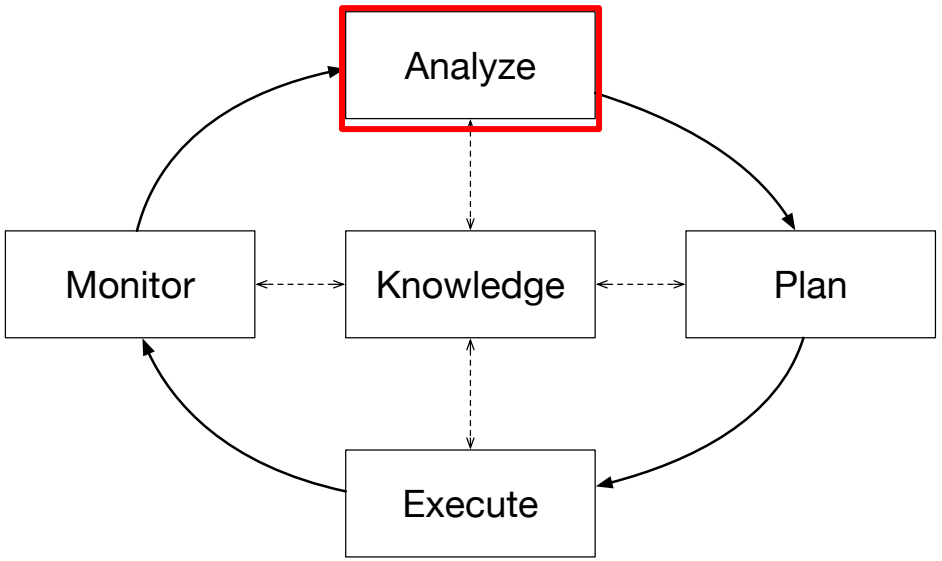


- Example:
  - Monitoring of network connection, user activity, weather conditions, location

MAPE-K
Adaptation control

- MAPE-K feedback loop as architectural reference for designing self-adaptive software systems

- M – Monitor
- A – Analyze
  - Evaluation of sensed values to determine if system needs to be changed
- P – Plan
- E – Execute
- K – Knowledge

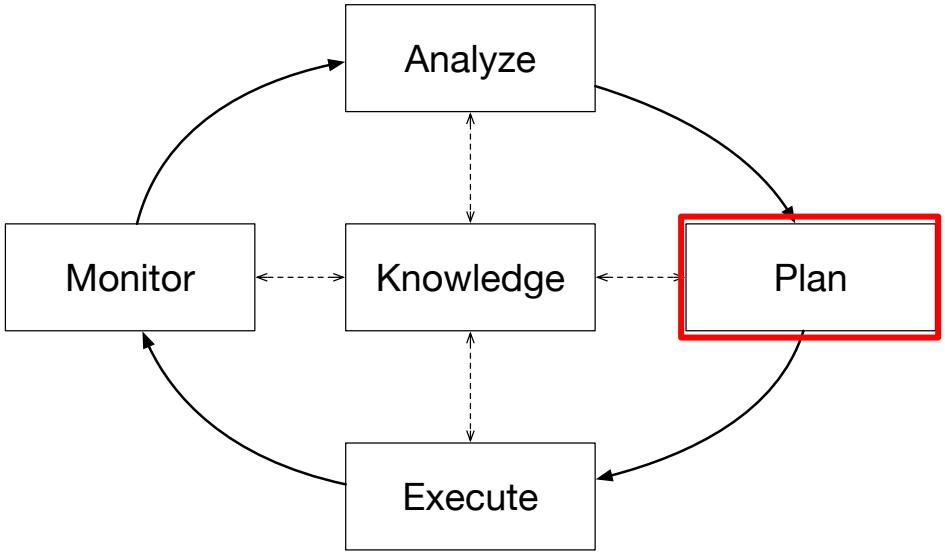


- Example:
  - Network connection lost AND re-establishment fails
  - ➢ Situation: Offline work

MAPE-K
Adaptation control

- MAPE-K feedback loop as architectural reference for designing self-adaptive software systems

- M – Monitor
- A – Analyze
- P – Plan
  - Reflecting on current system state and identifying appropriate state changes
  - Definition of adaptation plans
- E – Execute
- K – Knowledge

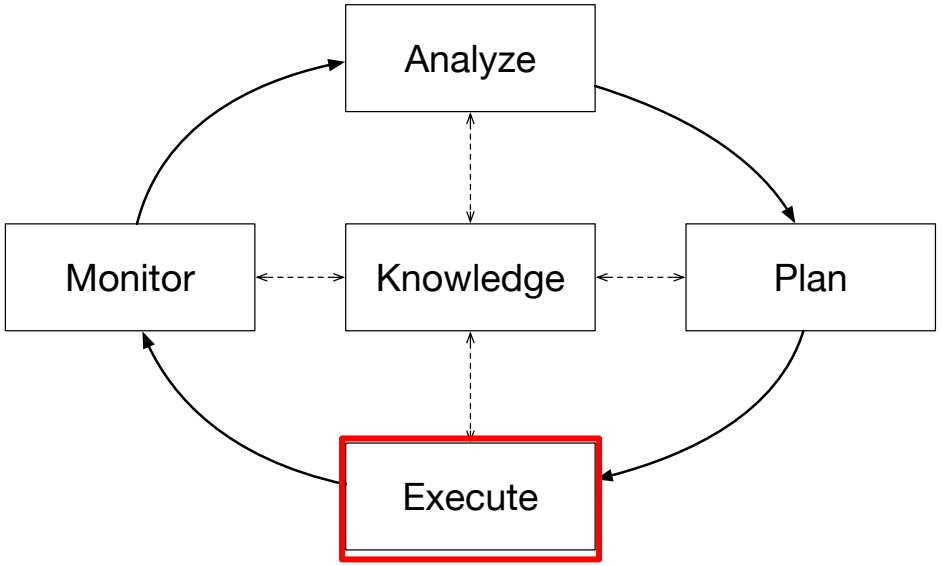


- Example:
  - Situation: Offline work
  - ➢ Establish local tracking, use cached data for timeline, activate queueing

MAPE-K
Adaptation control

- MAPE-K feedback loop as architectural reference for designing self-adaptive software systems

- M – Monitor
- A – Analyze
- P – Plan
- E – Execute
  - Execution of adaptation decisions on target system
  - Change of system configuration
- K – Knowledge

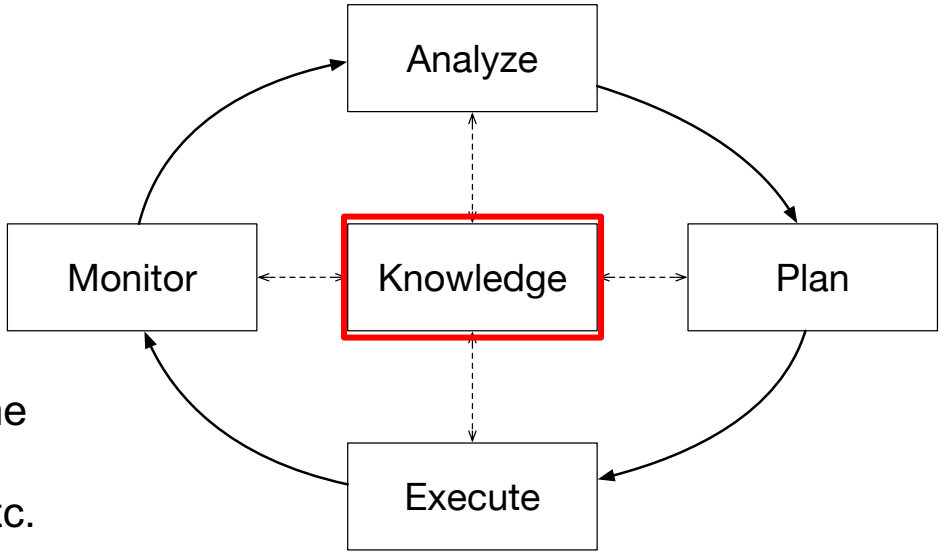


- Example:
  - Establish local tracking, use cached data for timeline, activate queueing
  - ➢ Execute parameter and structural changes

MAPE-K
Adaptation control

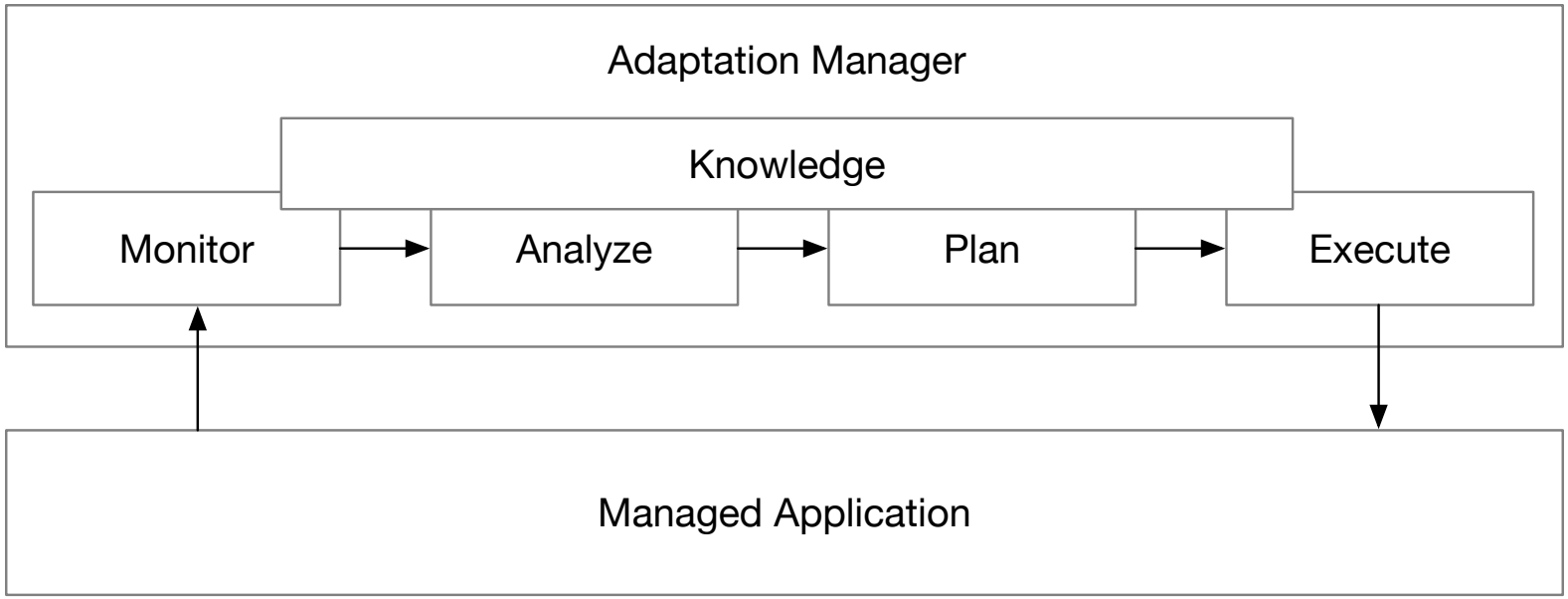
- MAPE-K feedback loop as architectural reference for designing self-adaptive software systems

- M – Monitor
- A – Analyze
- P – Plan
- E – Execute
- K – Knowledge
  - Stores information about the managed element, e.g., runtime or context model(s), history of collected sensor information etc.



- Example:
  - Update knowledgebase to reflect changed application configuration

Implementing Adaptation Control



- use of an external adaptation engine
- clear separation of concerns

Summary

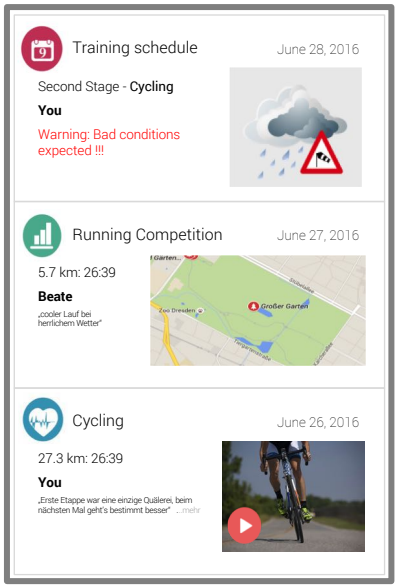
- Self-Adaptive Systems comprise
  - Context-Awareness, Self-Awareness and Self-Adaptation
- Context is pre-requisite for adaptation
  - Capturing and abstraction of context from heterogeneous sources
  - Situation detection by aggregating context values
- Adaptation mechanisms to adapt application data, data transmission and application structure
  - adaptation mechanisms operate on data or structure of application
  - ➢ self-awareness implemented in mechanisms
- Feedback loops to control adaptation process
  - Monitor, Analyse, Plan, Execute - Knowledge

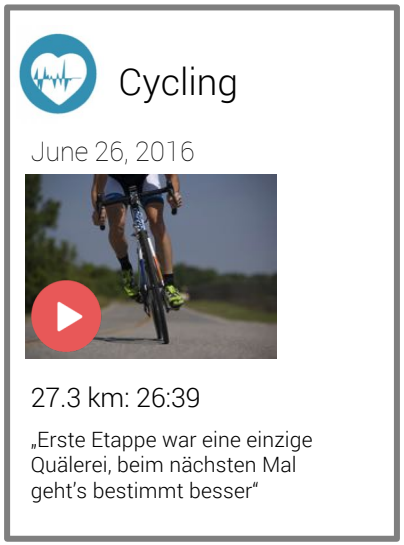
SEMINAR

Motivating Example

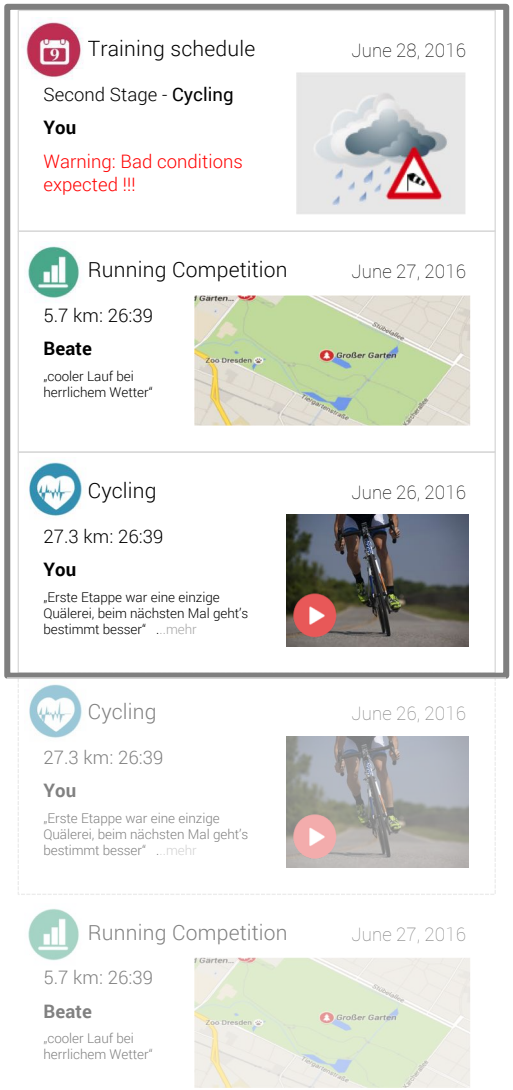
- Social Fitness App
  - Wristband/Smart Watch + Smartphone App
  - Server component for data storage and user management

- Functionality
  - Automatic detection and tracking of sports activities
  - Recording of activity states and content (images, videos, track record, curve with pulse, etc.)
  - **Activity Timeline -** Posting own activities and **see others‘ activities in an integrated timeline with text, images, videos, etc.**
  - Management of training schedule and planning of training activities
  - Managing competitions with ranking





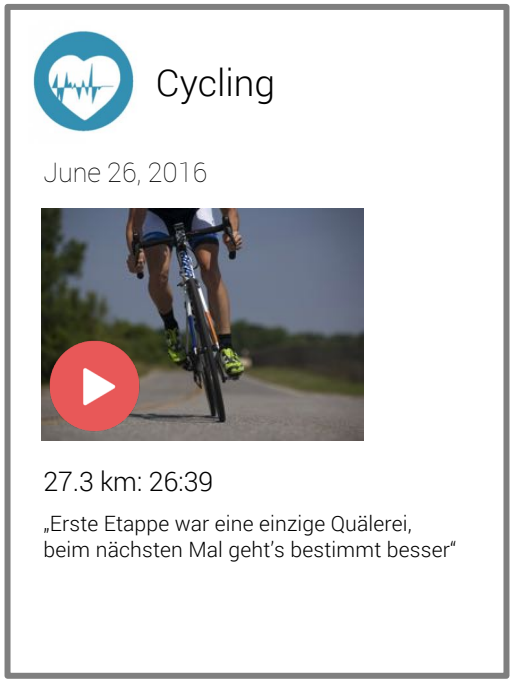
Prefetching vs. Lazy Evaluation



- How to use Prefetching/Lazy Evaluation in this scenario?

Detail View

Scrolling

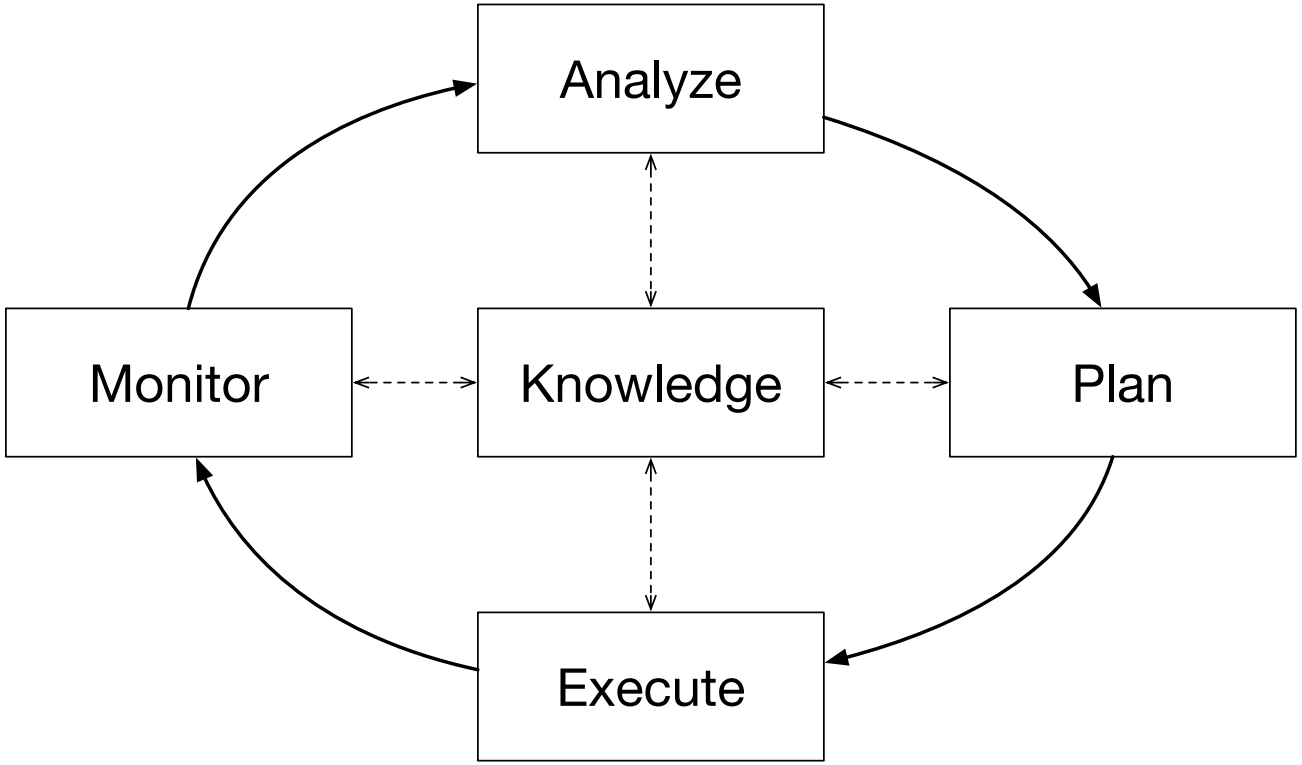


Prefetching vs. Lazy Evaluation

- Compare the two adaptation mechanisms Lazy Evaluation and Prefetching with respect to effect of data reduction and response time?
- What are Pros and Cons of Lazy Evaluation and Prefetching?
- What context is necessary to decide if Prefetching or Lazy Evaluation should be used?

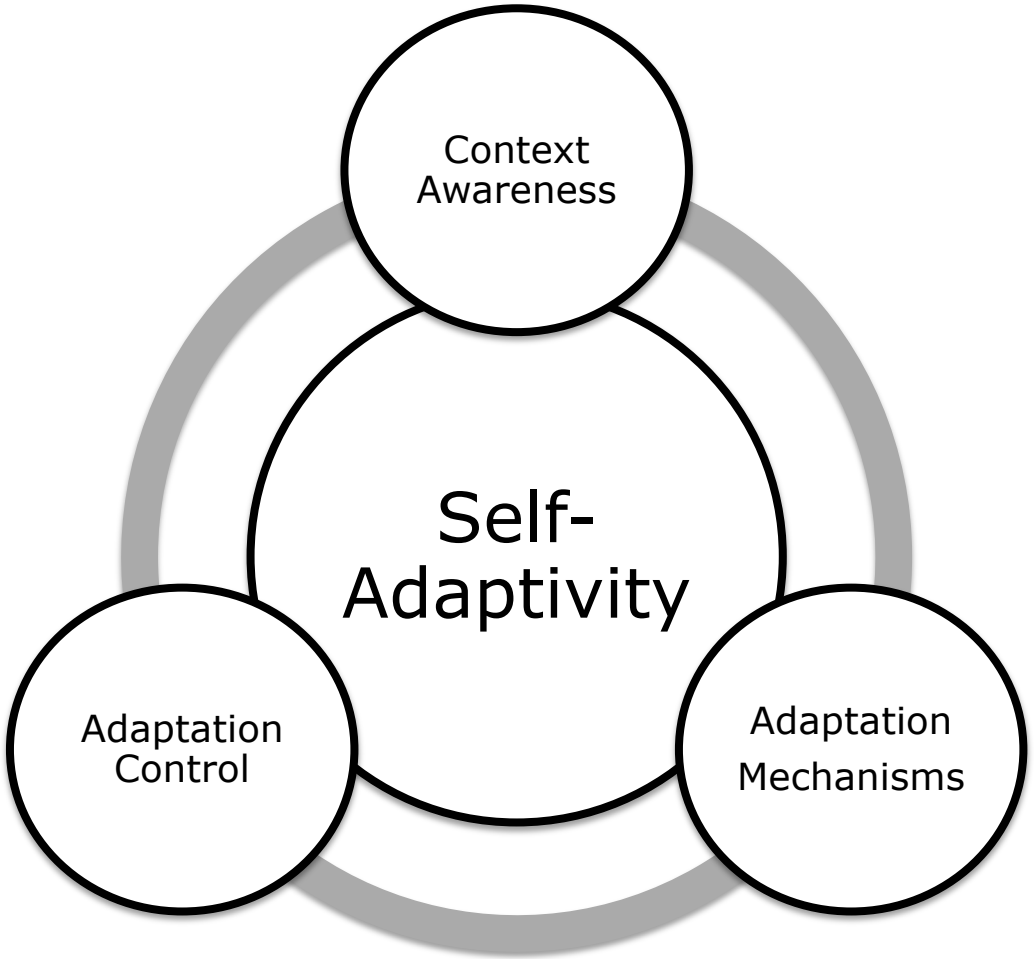
The MAPE-K Loop

- Explain the stages of a MAPE-K loop.
- What is the content of an adaptation plan?



Self-Adaptive Systems

- Which requirements have self-adaptive systems to fulfil?
- Explain the relation between context, adaptation mechanisms and adaptation control.



Further questions

- What is context?
- What role plays context for adaptive systems?
- What is the difference between Caching and Prefetching?
- When should filtering be used?
- How is the mechanism Queing working?