Distributed Realtime Systems Framework for
Sustainable Industry 4.0 applications

rialgo
realtime systems

March 2015

rialgo realtime systems GmbH | Robert-Bosch-Straße 7 | 64293 Darmstadt

Agenda

- Use case example
- Deterministic realtime systems
- Almost deterministic distributed realtime systems
- Distributed realtime systems framework
- Conclusion

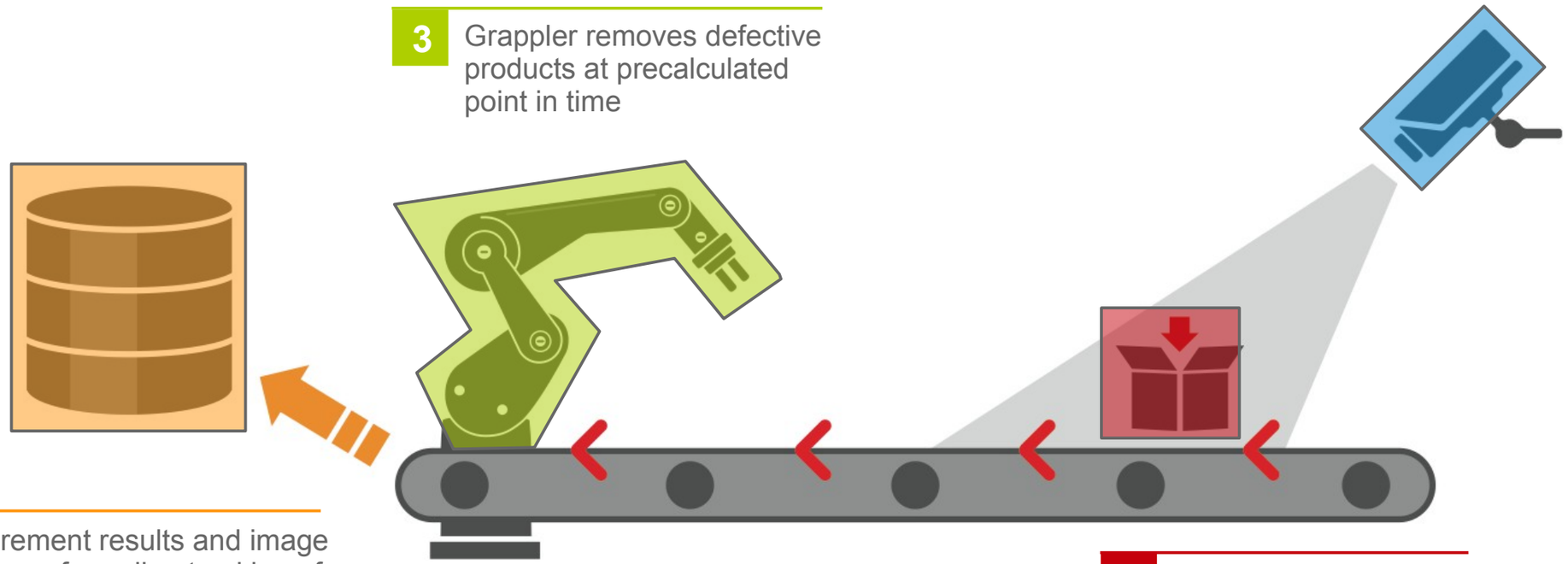
rialgo
realtime systems

Use case example

Automated defect sorting service

2 Camera observes belt conveyor and detects defective products in realtime

3 Grappler removes defective products at precalculated point in time



4 Export of measurement results and image data into a database for online tracking of product quality and deduction of actions to improve the manufacturing process

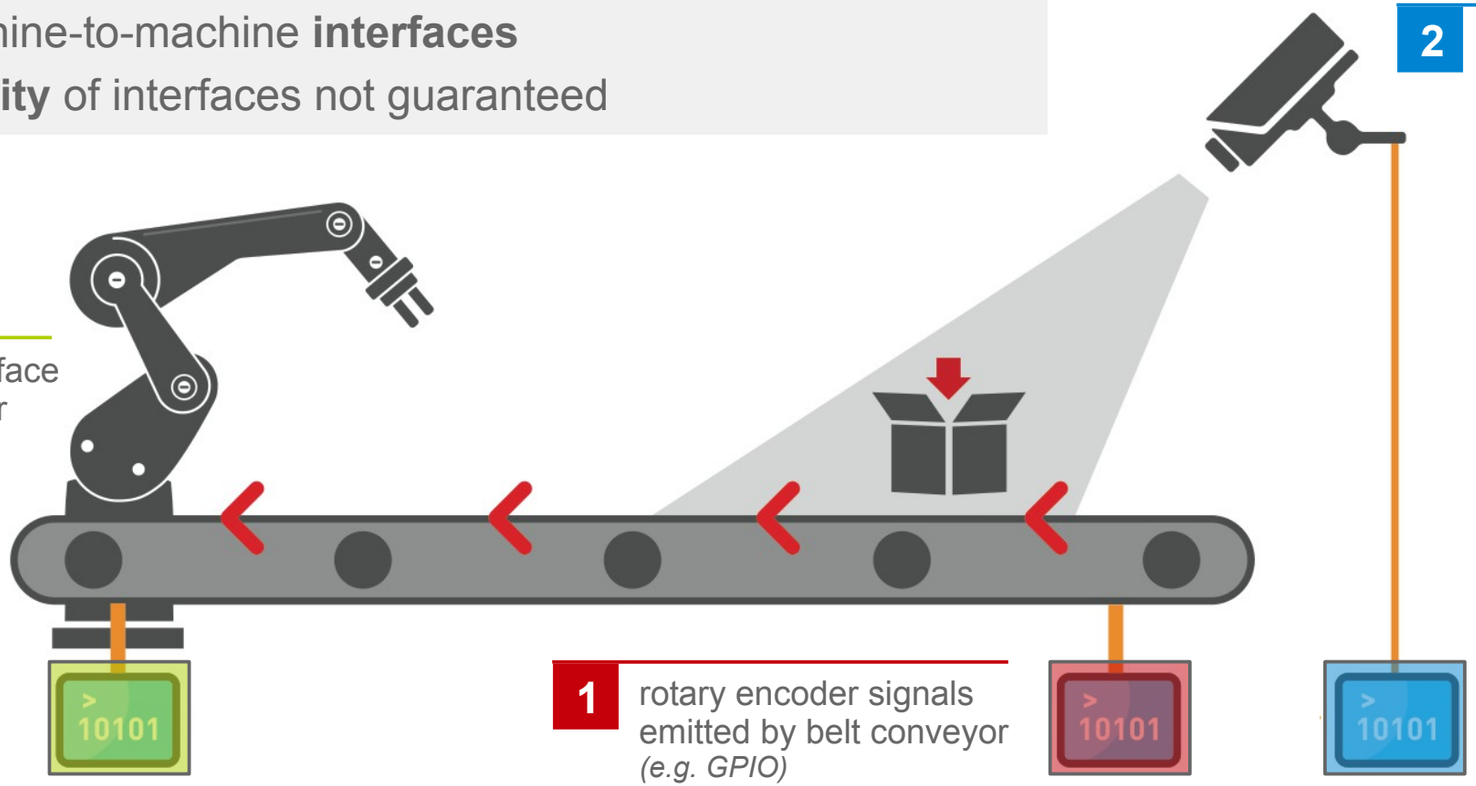
1 Products run through manufacturing process on a belt conveyor

Use case example

Automated defect sorting service

Equipment of production line with sensor- and actuator-components

- **Proprietary** machine-to-machine **interfaces**
- **Long-term stability** of interfaces not guaranteed



2 machine-to-machine interface for acquisition of image data *(e.g. GigE-Vision)*

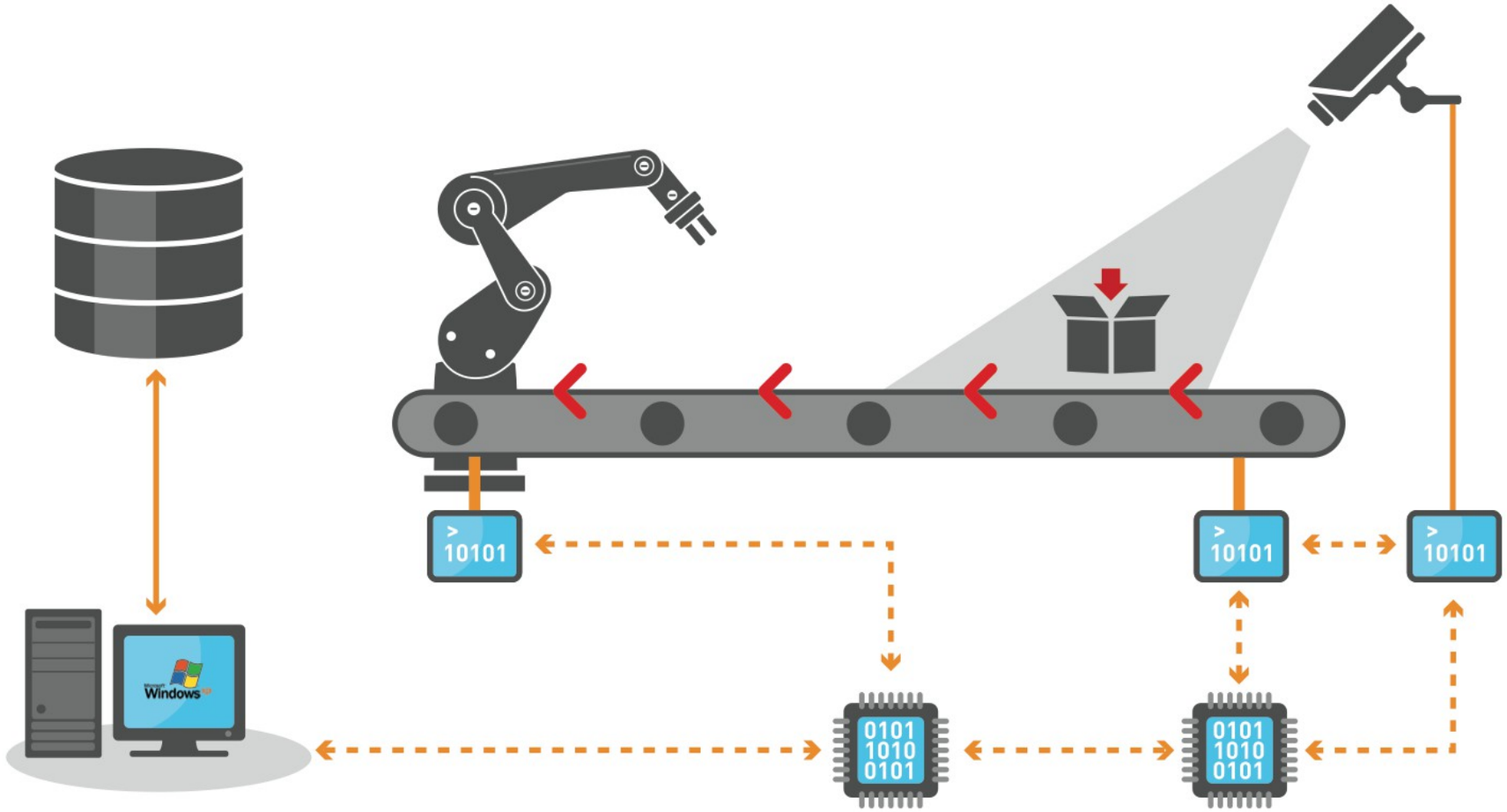
3 machine-to-machine interface for controlling the grappler *(e.g. serial port)*

1 rotary encoder signals emitted by belt conveyor *(e.g. GPIO)*

rialgo realtime systems

Today's realtime systems

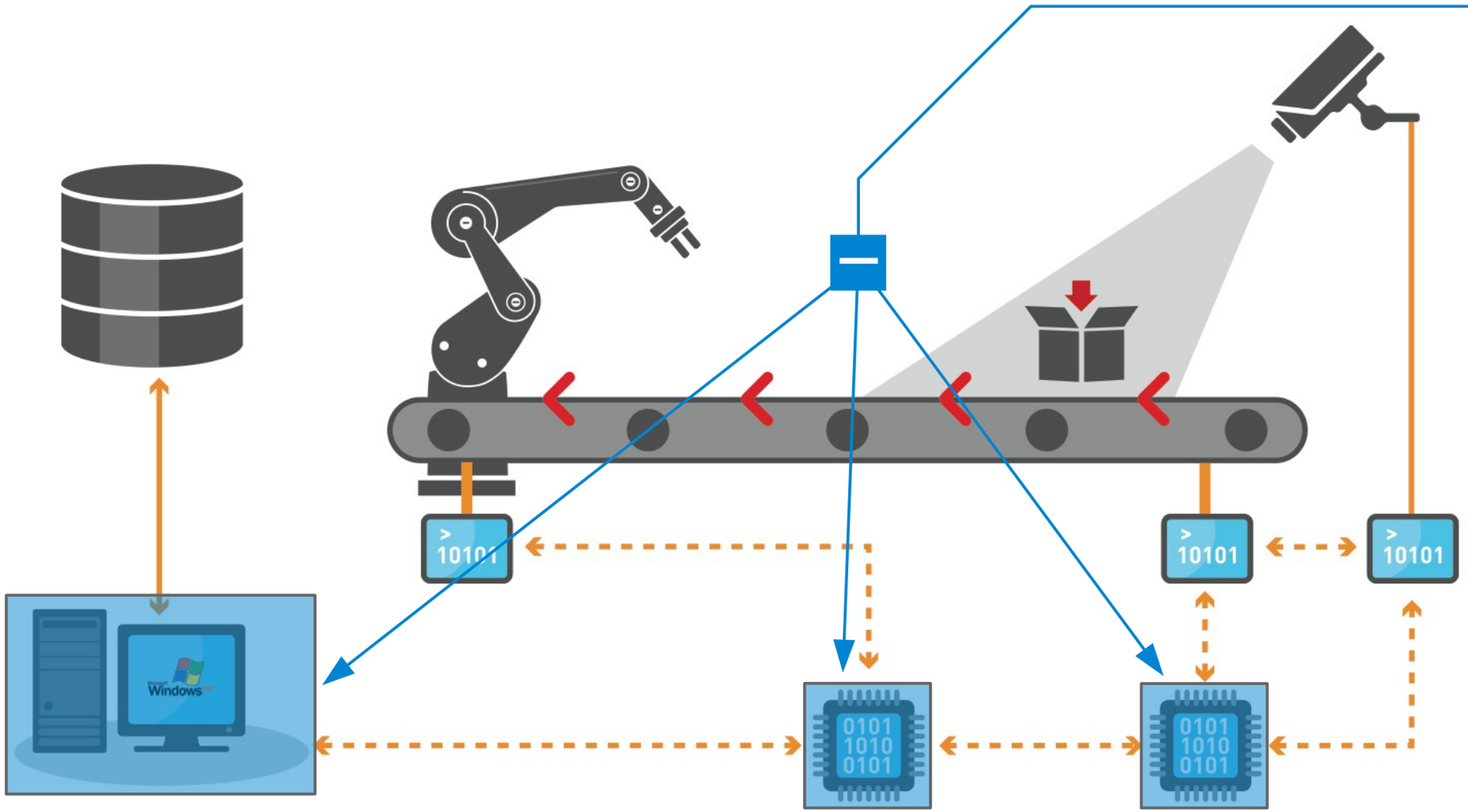
Usage of specialized hardware for realtime processing



rialgo realtime systems

Improper system design causes unsustainability

Deterministic run time behavior as main objective

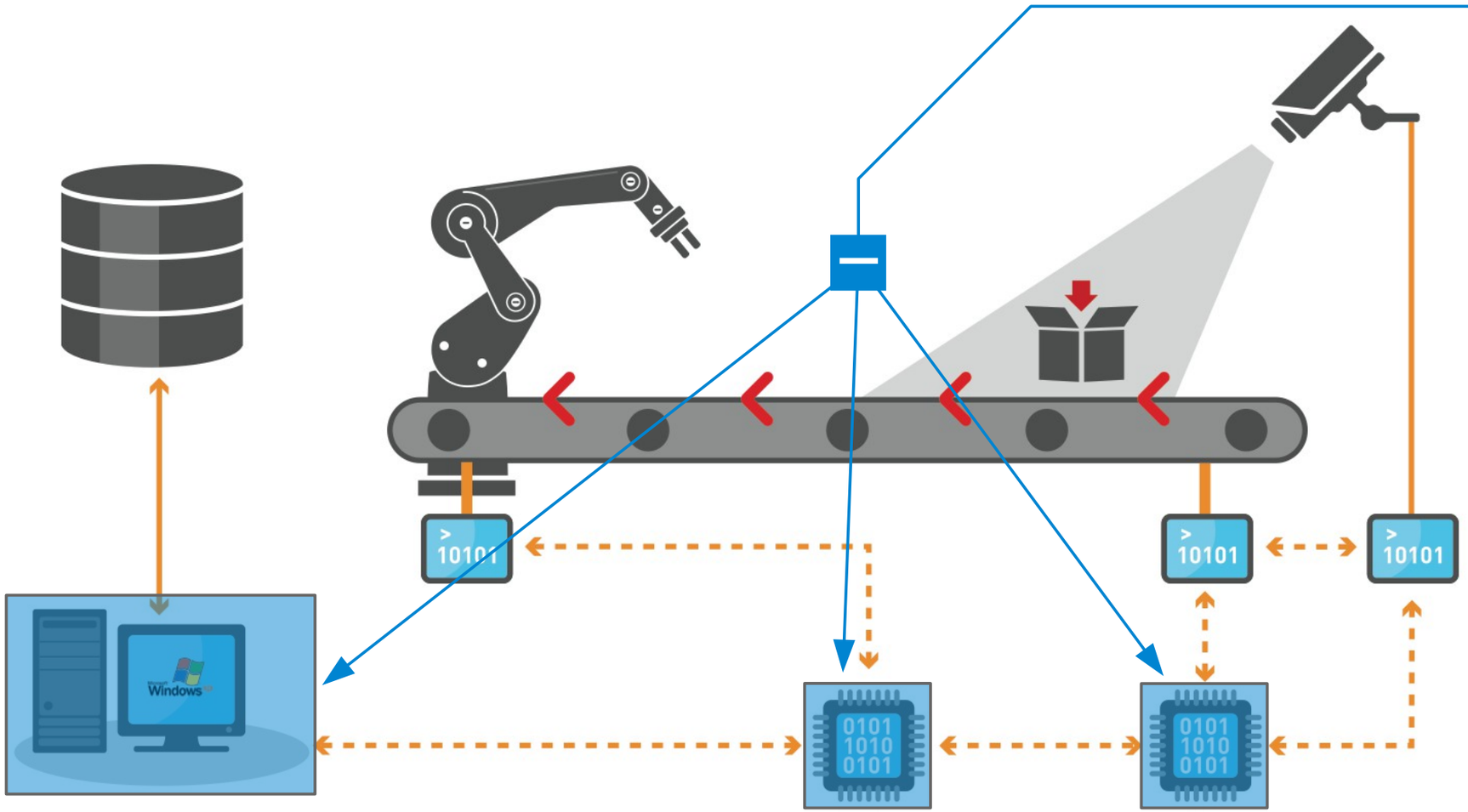


Strictly synchronized data processing

- **Monolithic system**
- Usage of specialized computing platforms with **time-limited availability** *(e.g. DSPs or FPGAs)*
- Almost no possibility for hardware upgrades due to **rigid interconnection between components**

Improper system design causes unsustainability

Runtime optimization of all software components

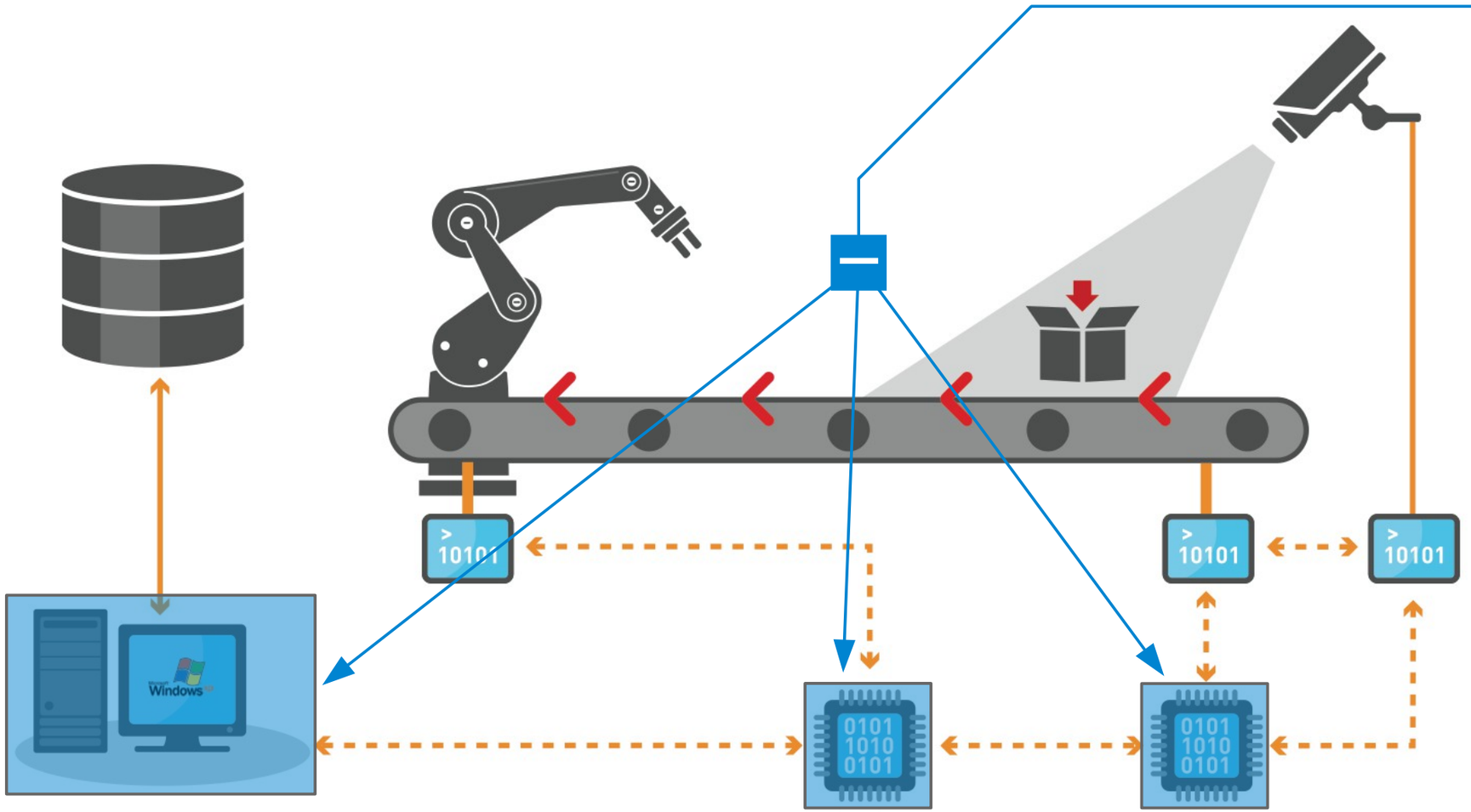


Execution speed at the expense of quality

- **Annihilation of best practices** for good software design
- Strong restraints on
  - **Maintainability**
  - **Reusability**
  - **Extensibility**
- **Few developers** with knowledge about functioning of complete system

Improper system design causes unsustainability

Deterministic realtime systems at any cost?

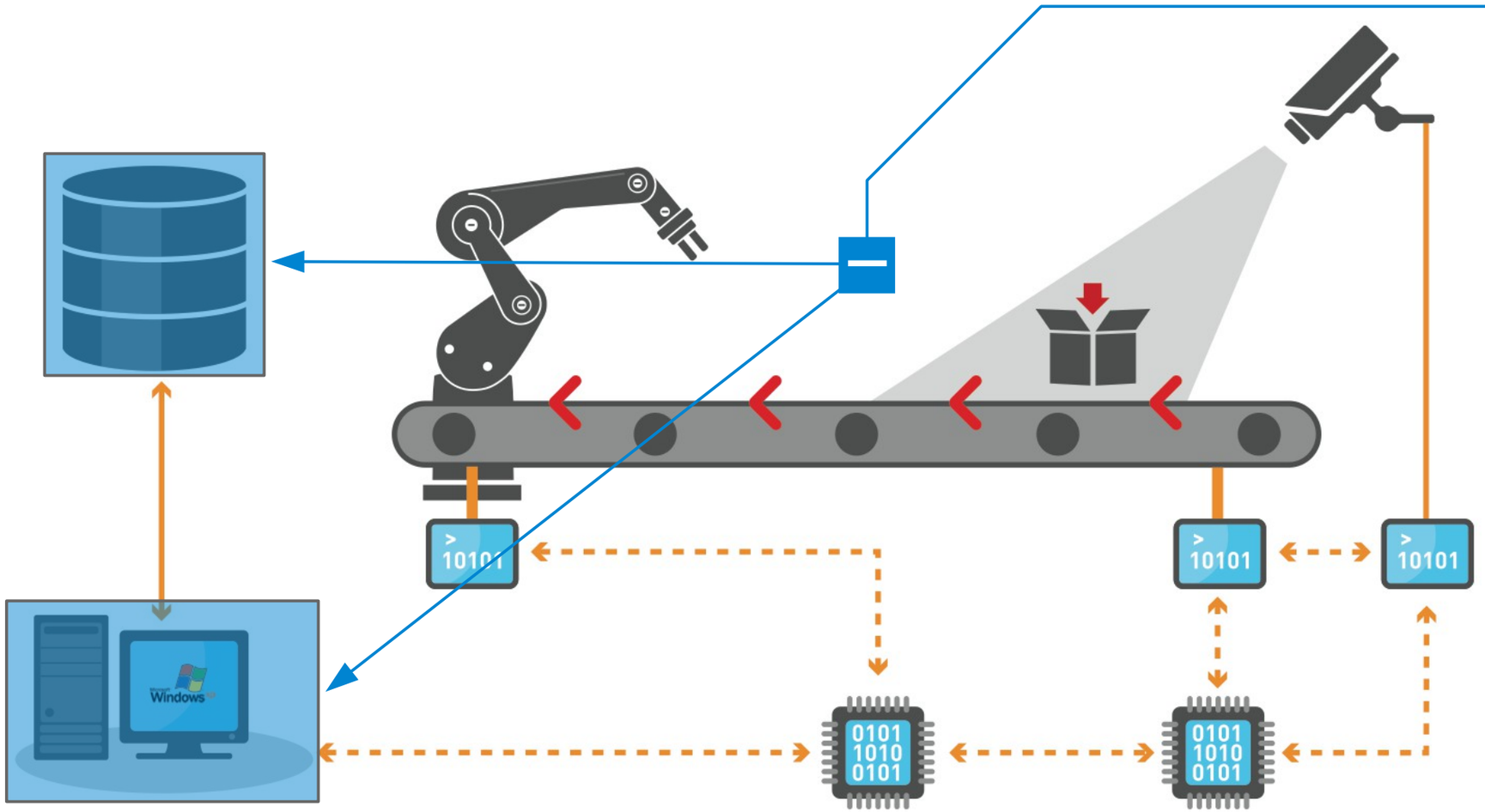


Expensive spare parts warehousing

- Long-lasting operation of **obsolete hardware platforms** for realtime processing
- Long-lasting operation of **obsolete master computers without availability of security updates** *(e.g. Windows XP)*

Improper system design causes unsustainability

Deterministic realtime systems at any cost?



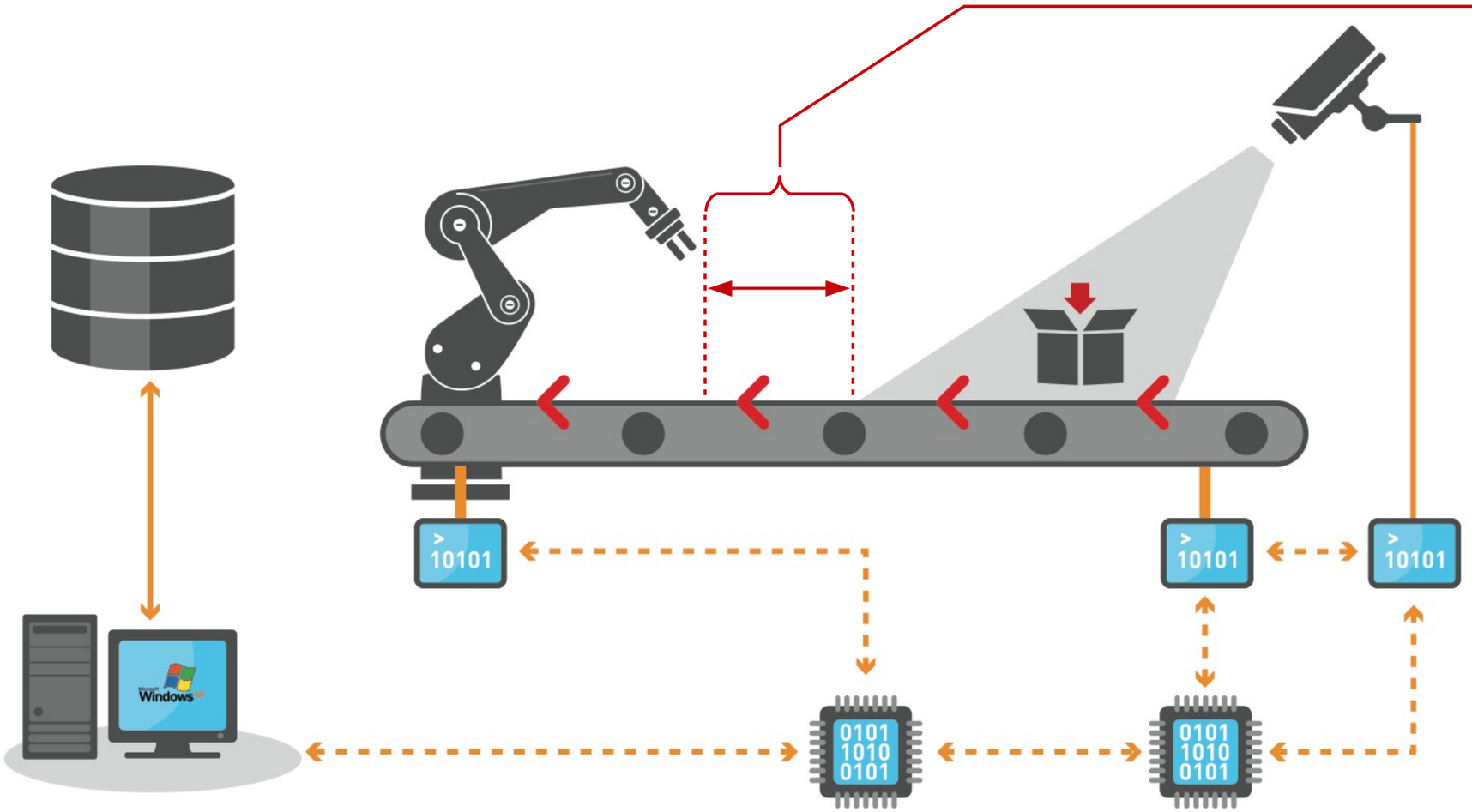
Updates cause high development costs

- **Incalculable implementation efforts** due to monolithic system design
  (e.g. rollout of new database technology)
- **Uncertain timelines for completion** due to small group of appropriate developers with in-depth system knowledge

rialgo realtime systems

Improper system design causes unsustainability

Are deterministic realtime systems always needed?



Time slots for runtime variations almost always existent!

- Runtime variation of optical inspection irrelevant if grappler receives out-sorting command **timely**
- Parameter for **scaling the time slot**
  - Velocity of the belt conveyor
  - Distance between camera and grappler

Improper system design causes unsustainability

Restraint in implementation of Industry 4.0 applications

Modern production lines		Today's realtime systems
Are capital goods that are used for a long period of time	⟷	Require warehousing of specialized hardware for long-lasting operation
Undergo changes due to continuous optimizations of the manufacturing process and short product innovation cycles	⟷	Cause incalculable implementation efforts for updates that require the availability of rare developers with in-depth system knowledge
Are critical company resources that need to be protected against illegal access	⟷	Cannot be easily updated to new OS-versions and represent a security risk as soon as the manufacturer support ends

Investment protection for Industry 4.0 facilities implies **planing reliability** und **cost transparency** for

- **Adaption of software components** to continuously evolving IT industry
- **Replacement of hardware components** by equivalent parts

Almost deterministic realtime systems

Idea

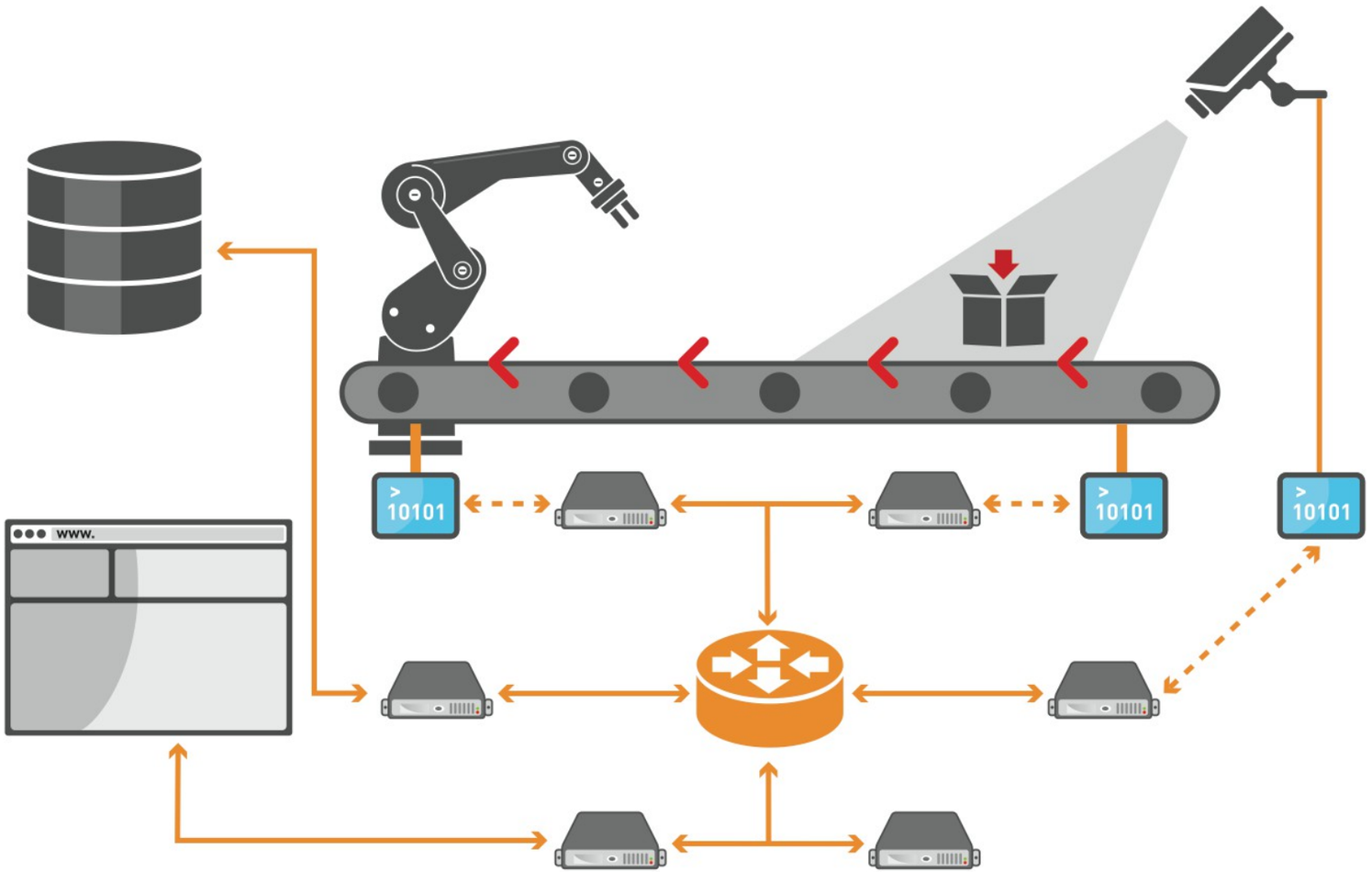
Use **off-the-shelf hardware** and **common multitask operation systems** to implement realtime systems, that are able to process incoming data **almost always fast enough** to derive **correct system reactions timely**.

Potential benefits of almost deterministic realtime systems

- OS as abstraction layer allows hardware upgrades
- IT security policies applicable to realtime systems
- Low software development costs due to widely accepted SDKs
- Large pool of appropriate software developers

Sustainability based on distributed system layout

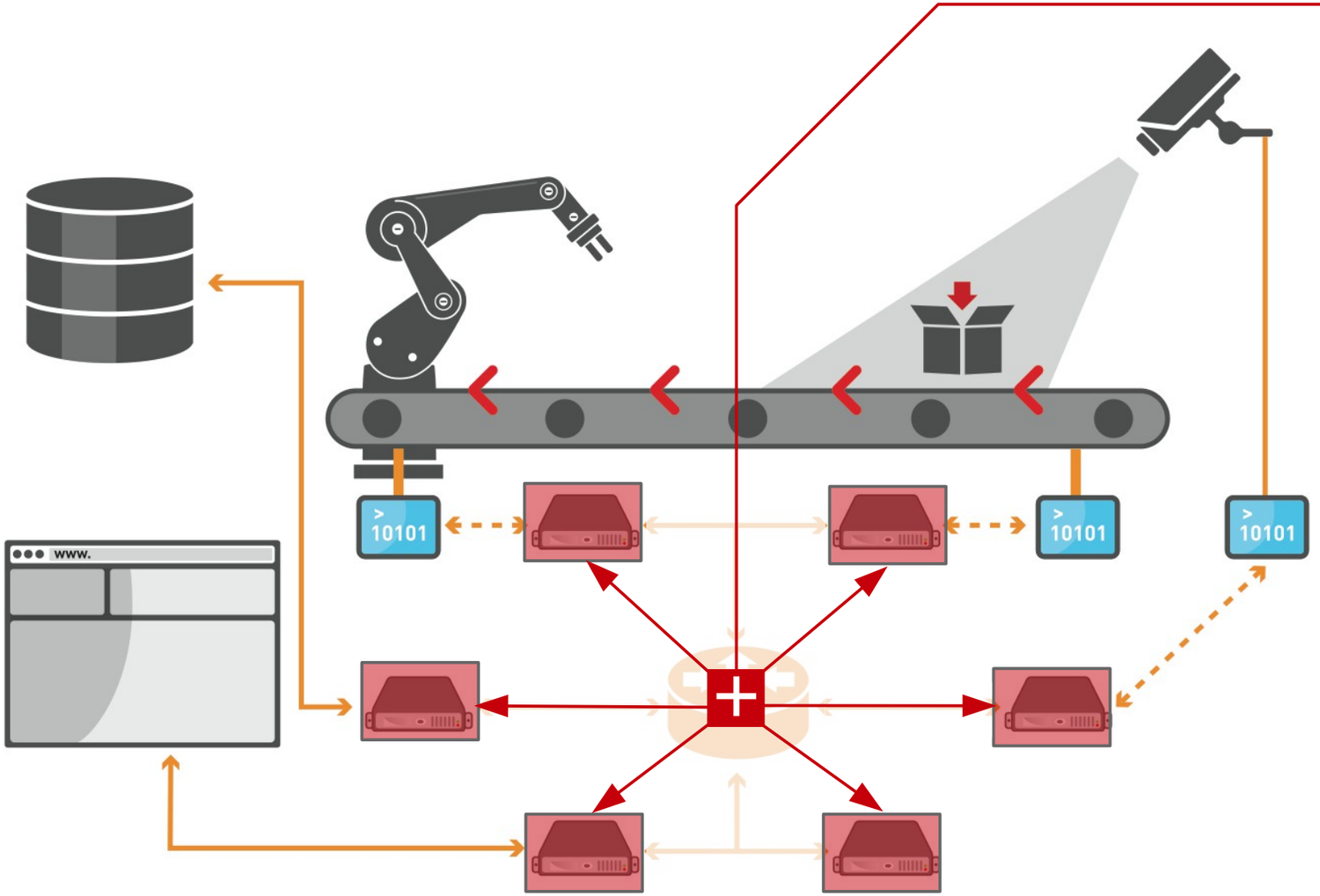
Combine approved IT components for almost deterministic realtime systems



rialgo
realtime systems

Sustainability based on distributed system layout

Off-the-shelf hardware for realtime processing



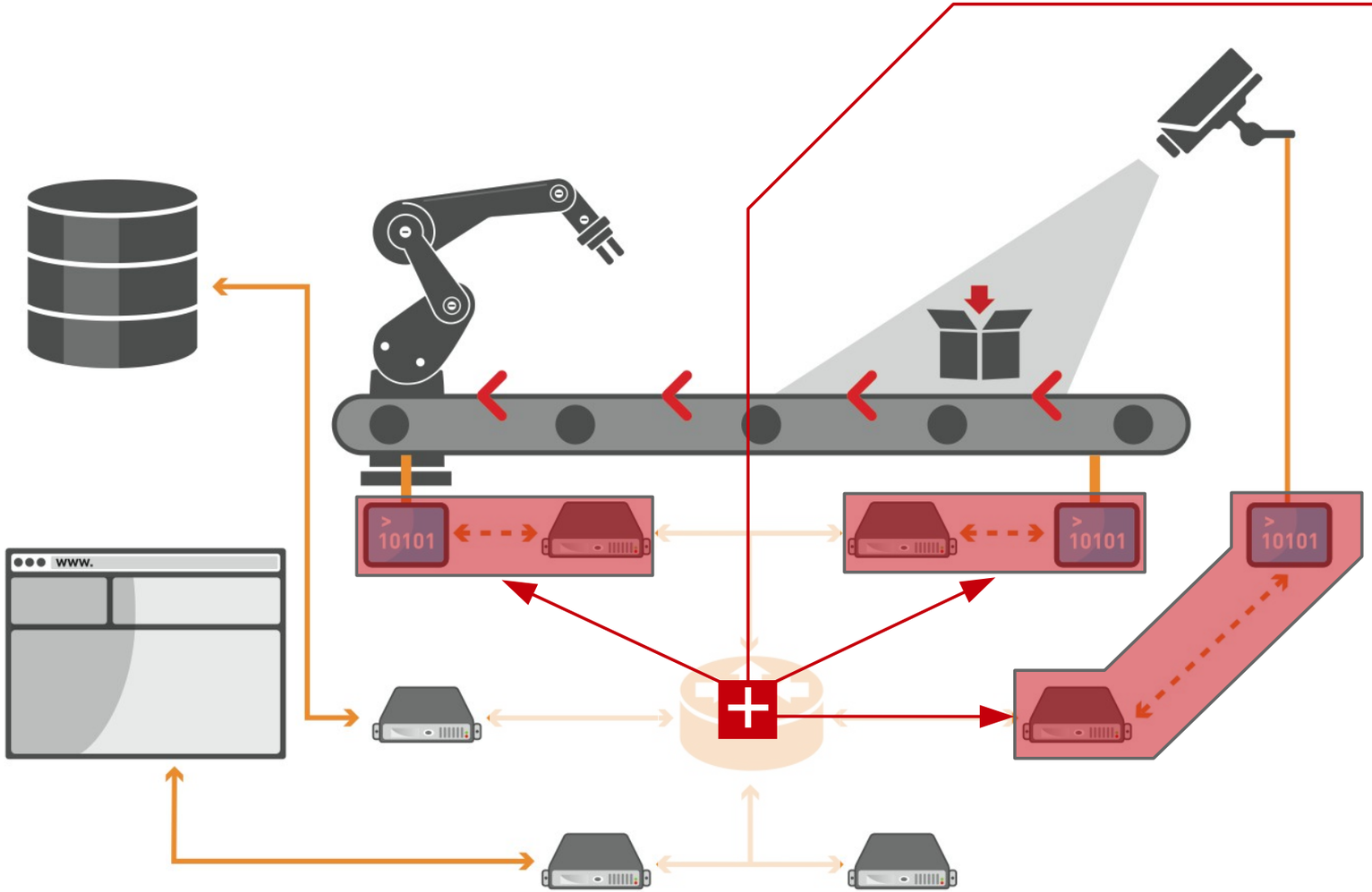
Server backend for realtime workflows

- **Division into subsystems** at host level
  - **Dedicated hosts** based on uniform hardware platforms
  - **Virtual hosts** deployed on uniform workstations
- **Uniform operation systems** with small deviation from default distributions

rialgo realtime systems

Sustainability based on distributed system layout

Uniform integration of specialized hardware

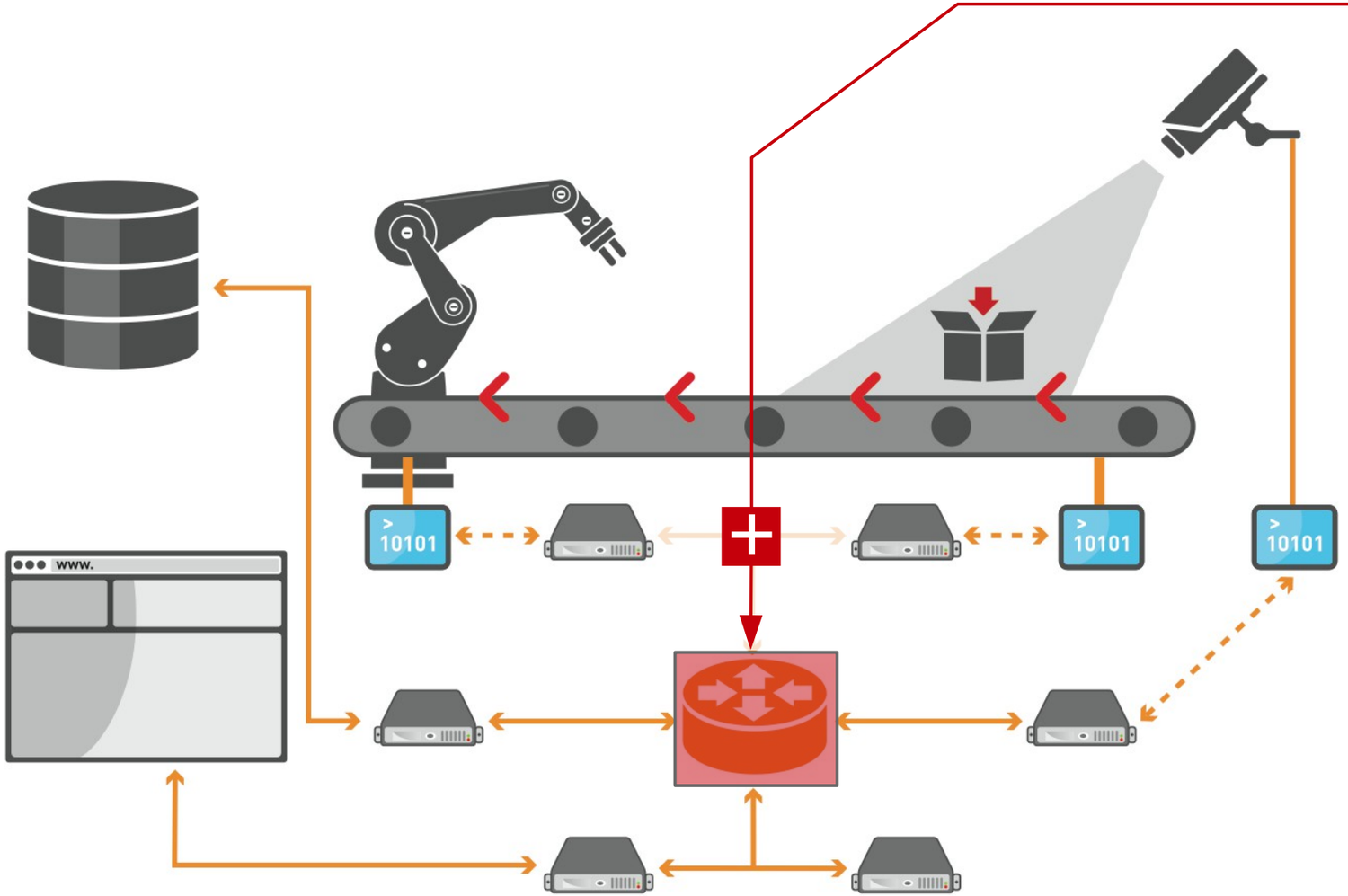


Abstraction of specialized hardware

- **Dedicated hosts** for integration of sensor- and actuator components
- **Manageable effort** for replacing specialized hardware without side effects on overall systems
- **Ability for incremental updates** at overall system level
  (e.g. integration of camera with increased resolution)

Sustainability based on distributed system layout

Common interfaces for linkage of subsystems



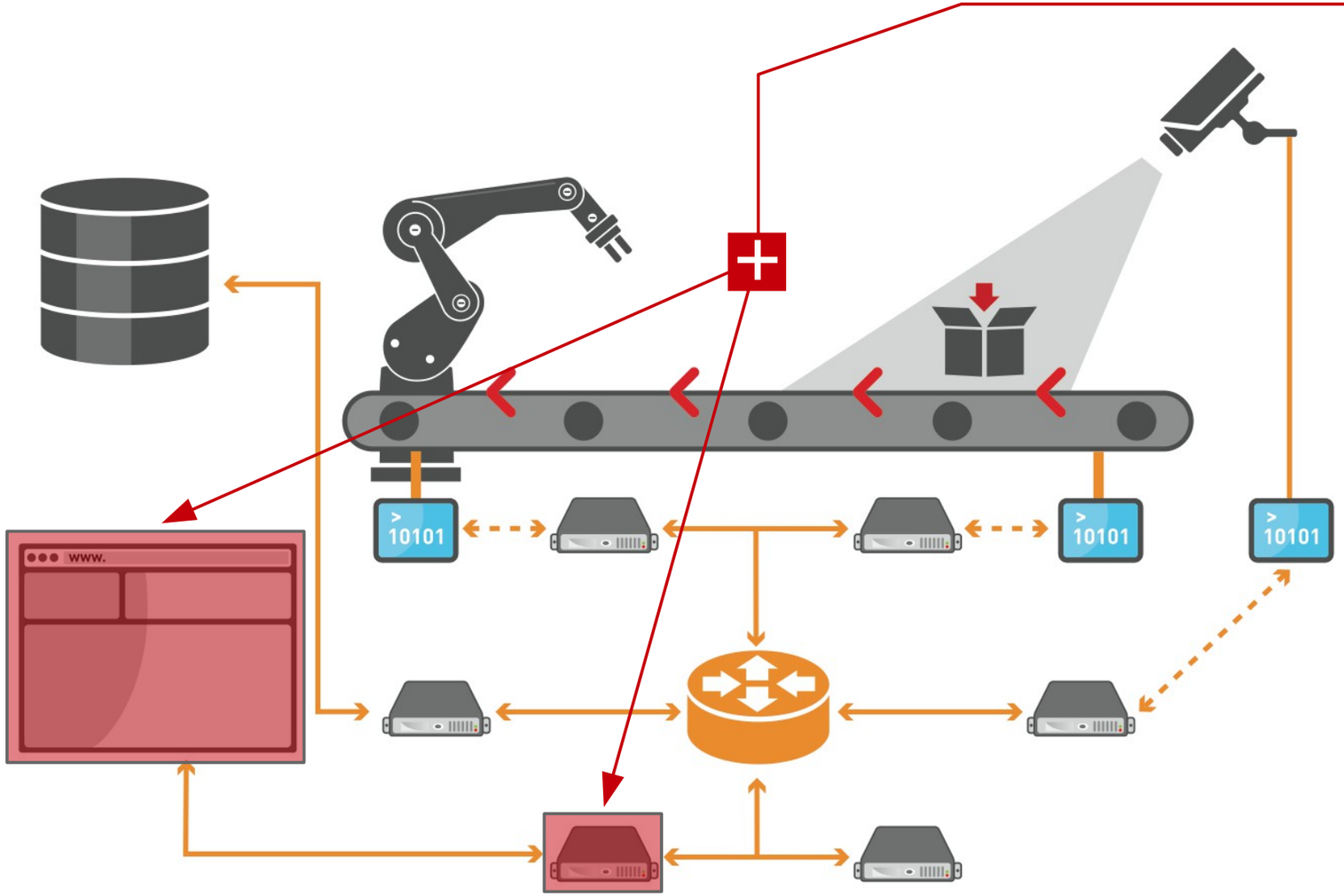
TCP/IP connectivity for data exchange

- **Future-proof** network infrastructure allows
  - Physical separation
    (e.g. housing of computationally intensive subsystems in cooled racks)
  - Expansion of additional hosts
- Utilization of industry standards for
  - **Time synchronization**
  - **Protection** of realtime system against illegal access

rialgo realtime systems

Sustainability based on distributed system layout

Web technologies for device independent HMI

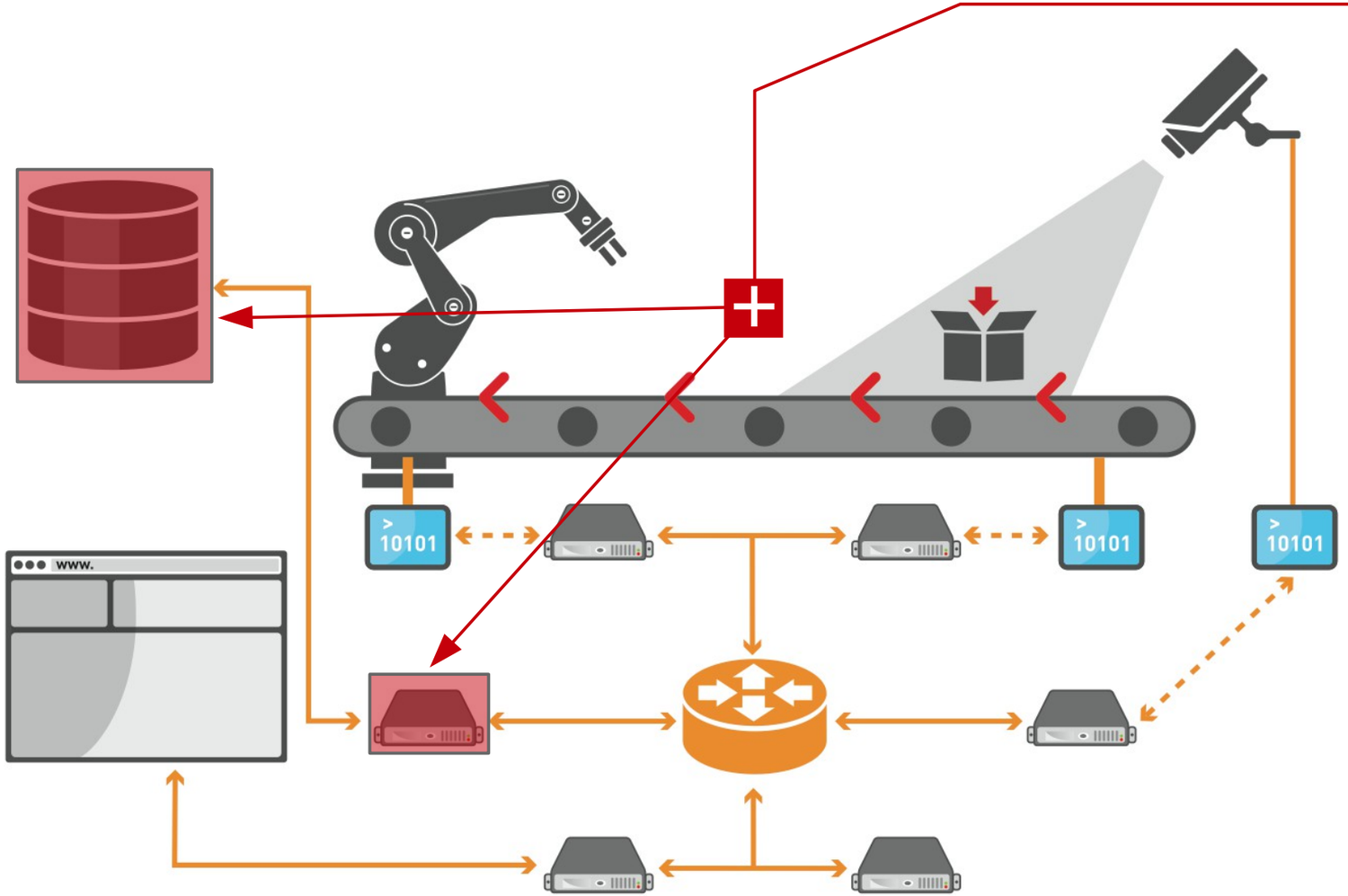


State of the art HMI for realtime systems

- **Dedicated host** provides web interface
- **Manageable efforts for updates** due to strict separation between HMI and processing hosts
  (e.g. adaption on new browser version)
- **Role based access restrictions** on configuration and data of the overall system

Sustainability based on distributed system layout

Database as output interface of realtime processing



Provision of processing results

- Database allows access to results **without interference** on realtime system
- Connection between database and realtime processing workflow via dedicated host
- Abstraction of underlying technology ensures **manageable effort for changes on database system**

rialgo realtime systems

Almost deterministic distributed realtime systems

Challenges in developing distributed realtime systems

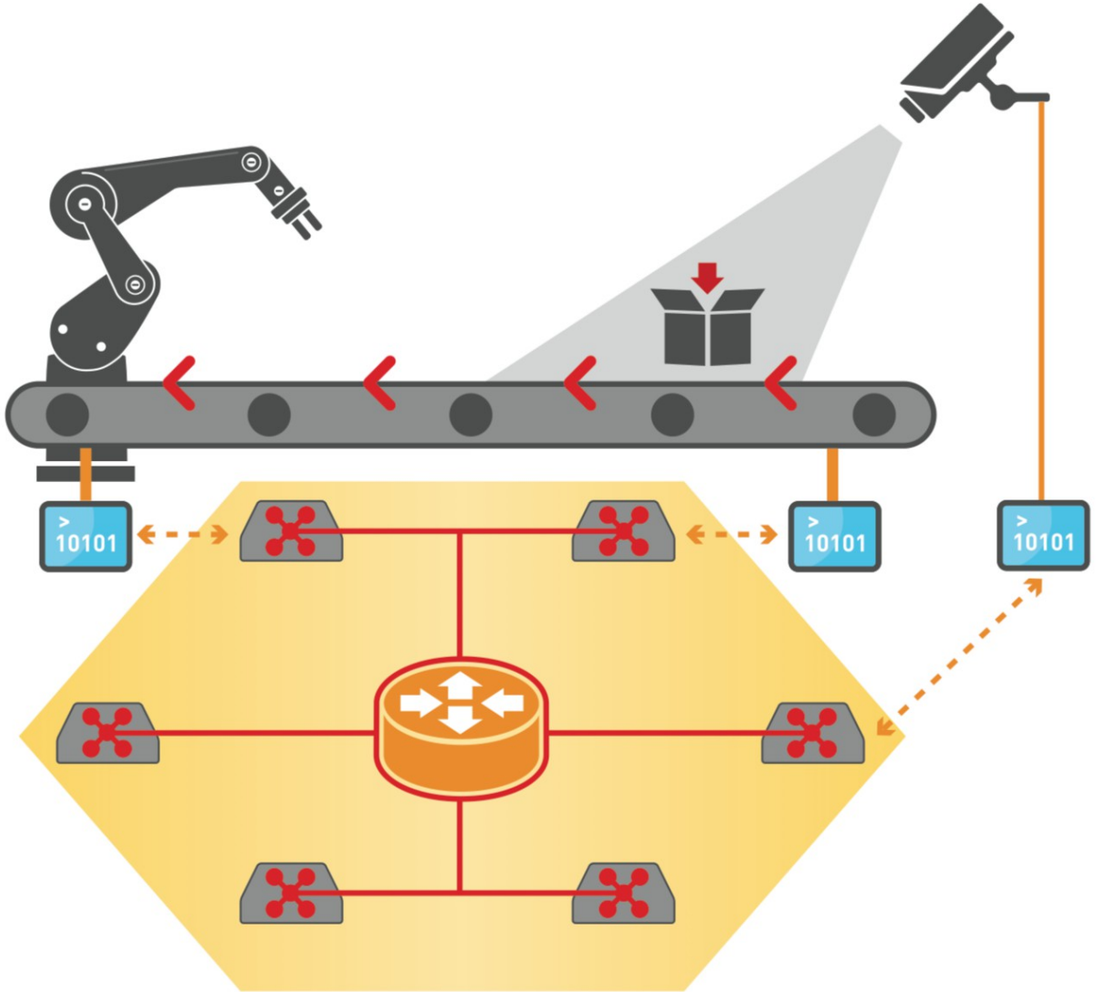
- ! Efficient mechanisms for mass data exchange
- ! Inherent asynchronism
- ! Generic mechanisms for installation, configuration & deployment

Goal

Provide a **framework**, that shows **almost no operating-system-specific routines** and enables the development of **highly efficient, distributed and long-term stable** realtime systems.

Distributed Realtime Systems Framework (DRSF)

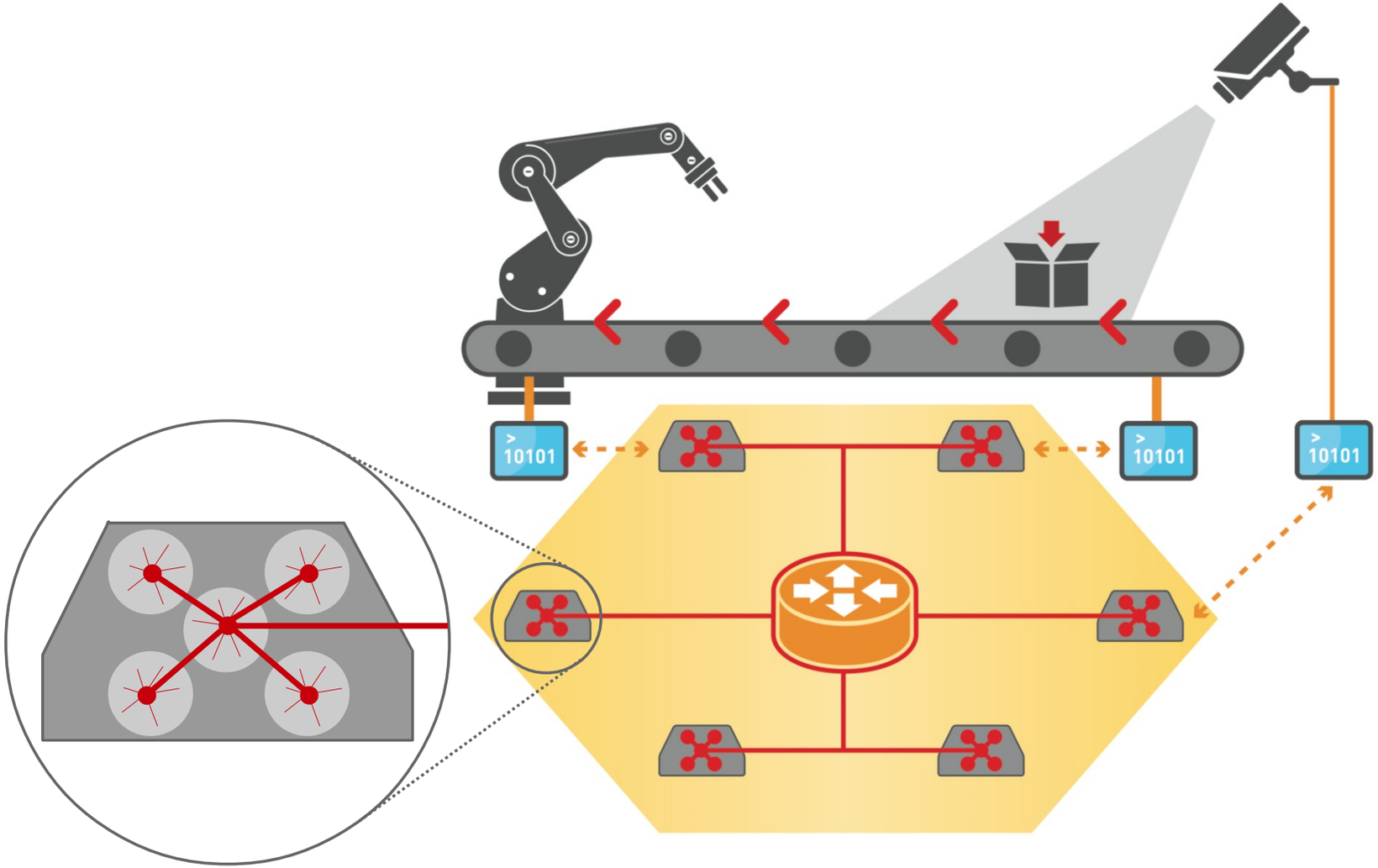
Distributed Realtime Messaging Framework (DRMF)



Uniform interprocess communication for (mass) data exchange

- Utilization of **commonly available media** for interprocess communication
  - Shared memory
  - TCP/IP
  - File system
- **Abstract all media** and provide highly efficient messaging channels across all hosts
  - Turn cluster of hosts into **one logical device** for realtime processing
  - Independence of underlying hardware
- **Highly available messaging channel**
  - Decentralized management
  - Robustness against failures
- **Sustainable software design**
  - No OS-specific routines
  - Designed for multithreaded operation

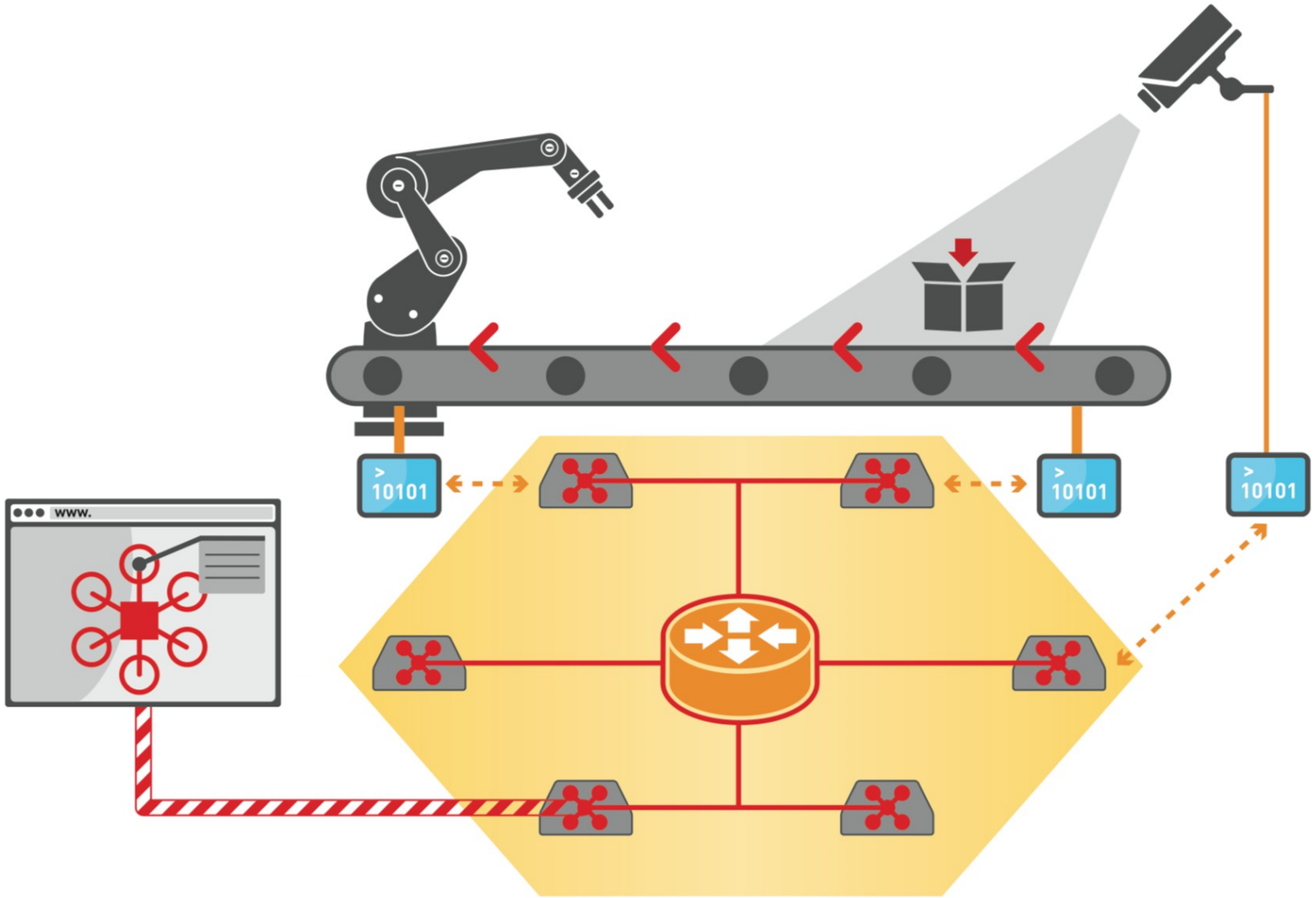
Distributed Realtime Application Framework (DRAF)



Host internal segmentation of subsystems at process level

- Construct modular programs that operate on messaging channels to **implement reusable functionality**
  - Almost **no latency for access on internal messaging channels** due to utilization of shared-Memory
  - Distribution of **realtime workflows on multi-core CPUs** due to scheduling of numerous modules on one host
- Construct subsystems by **linking modules with messaging channels**
- **Sustainable software design**
  - No OS-specific routines
  - Encapsulate asynchronous access on messaging channels to ensure controllability by a wide range of developers

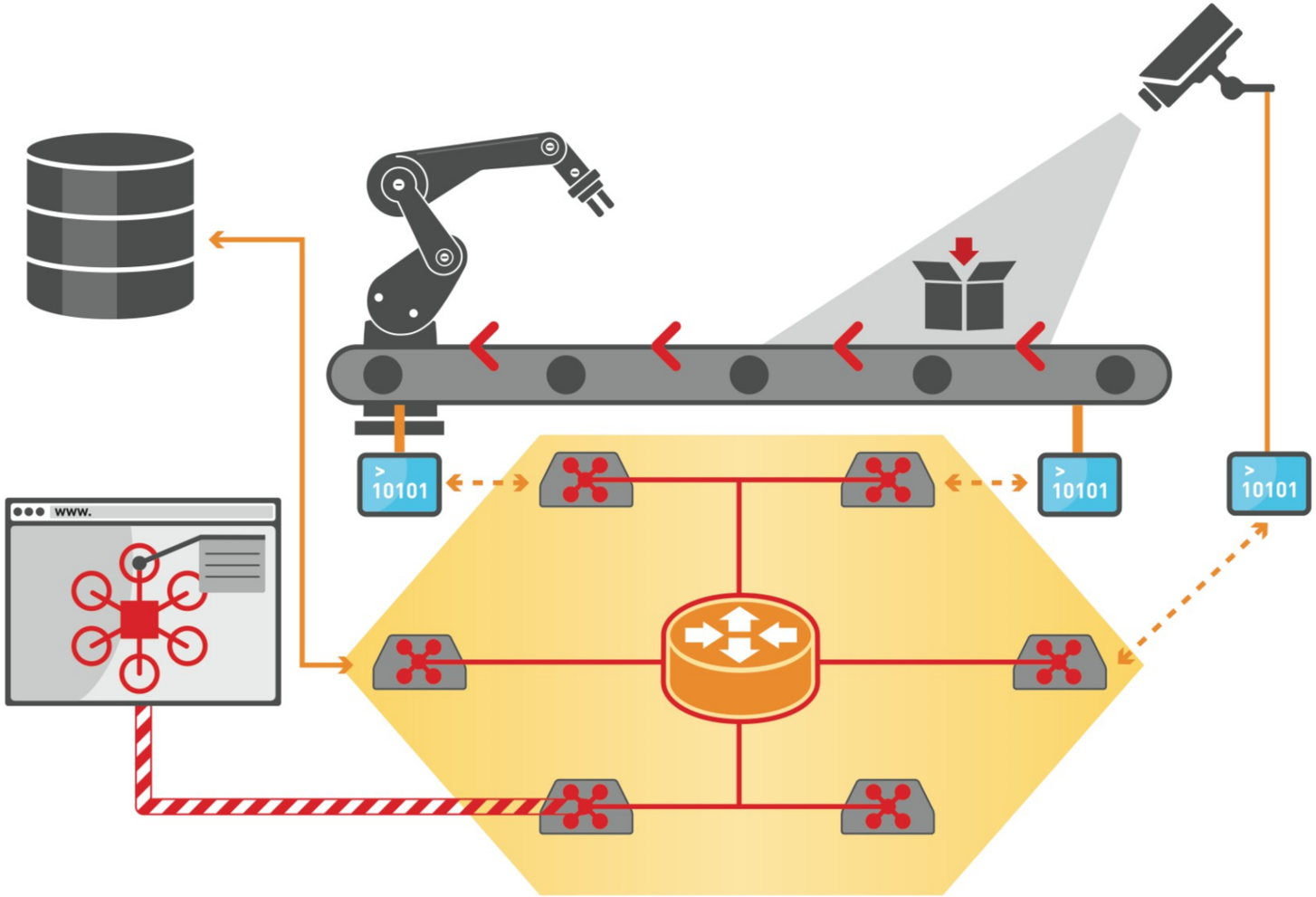
Distributed Realtime Application Web-interface (DRAW)



Intuitive device independent HMI

- Utilization of state of the art web technologies for ubiquitous HMI
- **Installation, configuration & monitoring** of complex workflows
  - Add & remove modules at runtime
  - Online visualization of messaging channels
- HMI based on **uniform interfaces** for
  - Configuration of messaging channels → **DRMF**
  - Linkage of modules with channels → **DRAF**
- **Sustainable software design**
  - Strict separation between HMI and realtime processing
  - Periodic updates on HMI to incorporate new web technologies without side effects

Database as media for interprocess communication

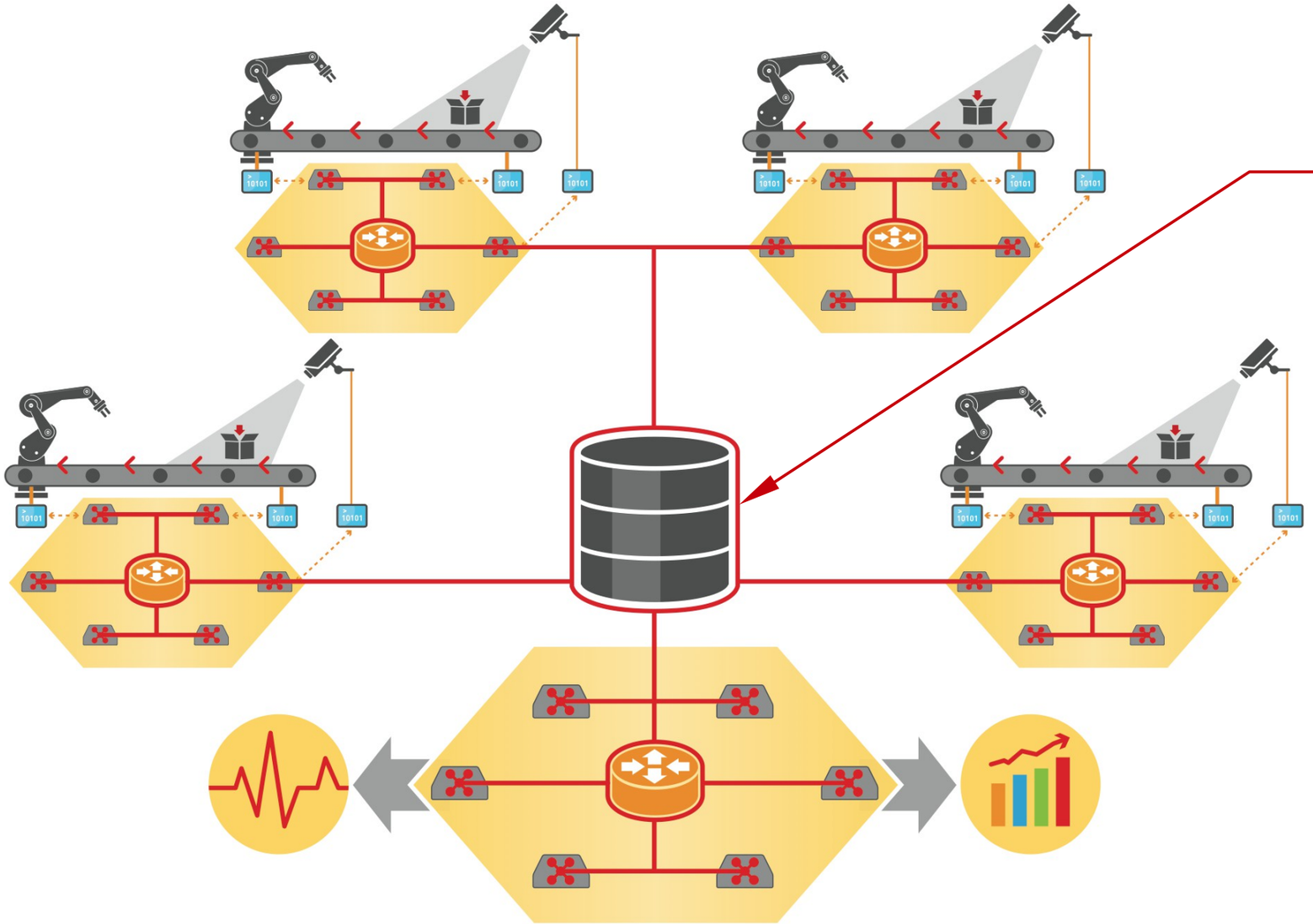


Integration of realtime systems into corporate IT infrastructure

- Mapping between realtime messaging channels and database structures
- Utilize database to establish **interprocess communication at corporate level**
  - High performance systems for **online big data analysis tasks** *(e.g. SAG Terracotta BigMemory)*
  - Off-the-shelf systems for data storage and subsequent offline analysis tasks *(e.g. MySQL)*
- **Sustainable software design**
  - Abstraction of database technology
  - Manageable effort for updates and rollout of new technologies

Distributed Realtime Systems Framework

Company-wide big data analysis



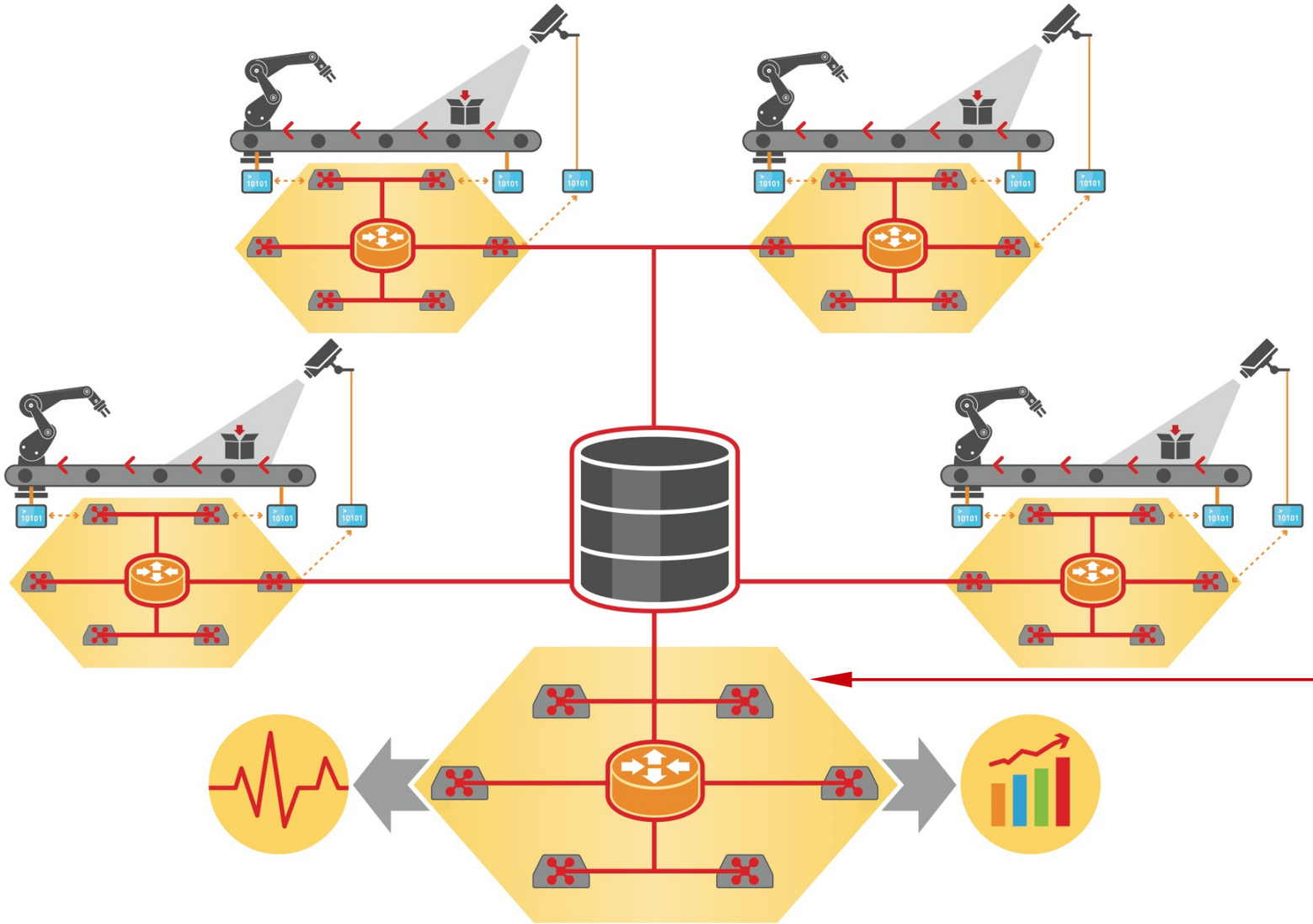
Database as corporate connector between distinct realtime systems

Realtime messaging channels are stretched across high performance database
*(e.G. **SAG Terracotta BigMemory**)*

rialgo realtime systems

Distributed Realtime Systems Framework

Company-wide big data analysis

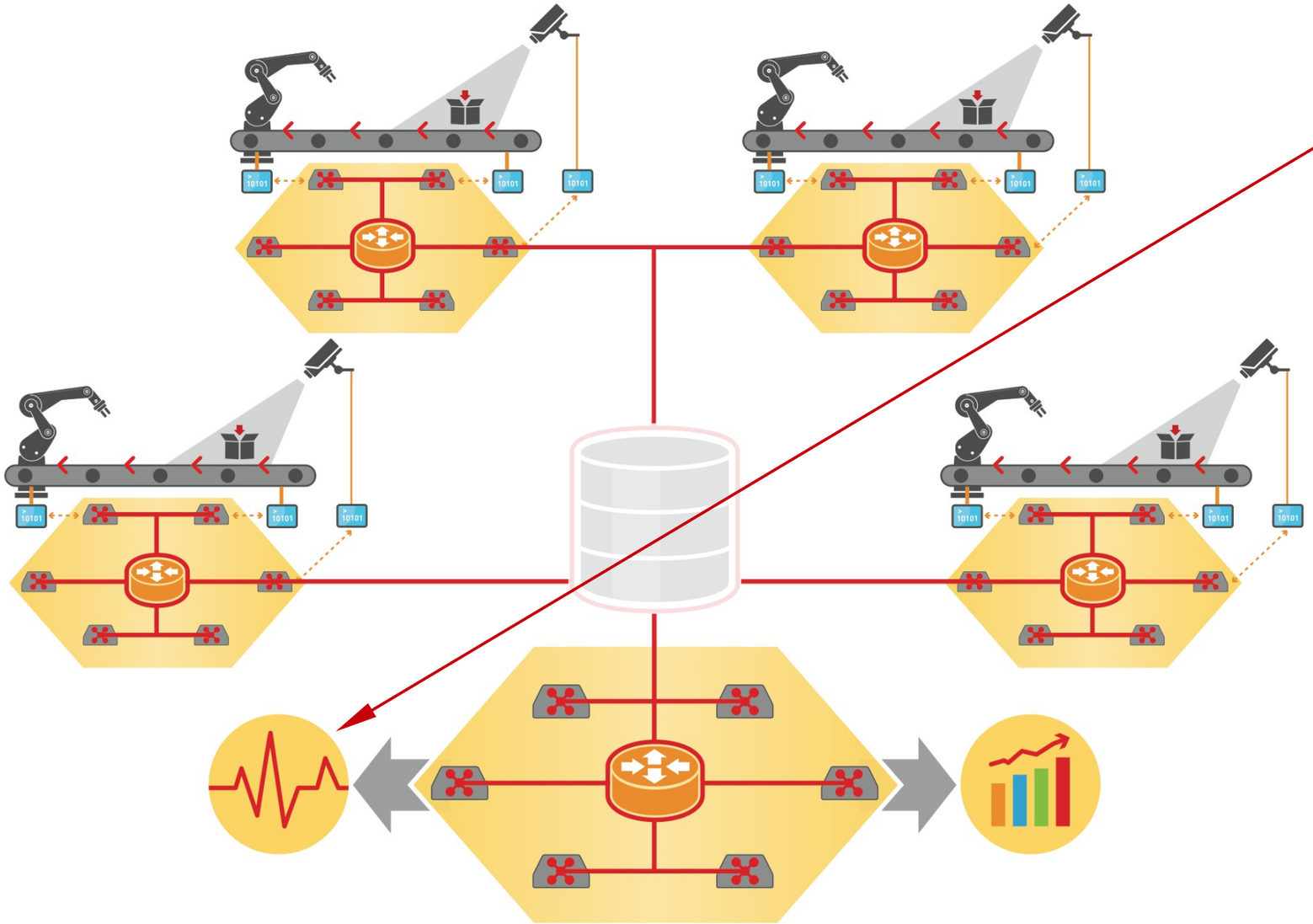


Distinct realtime system for online big data analysis

Development of sustainable algorithms subdivided into numerous reusable programs without direct references to realtime systems of production lines

Distributed Realtime Systems Framework

Company-wide big data analysis

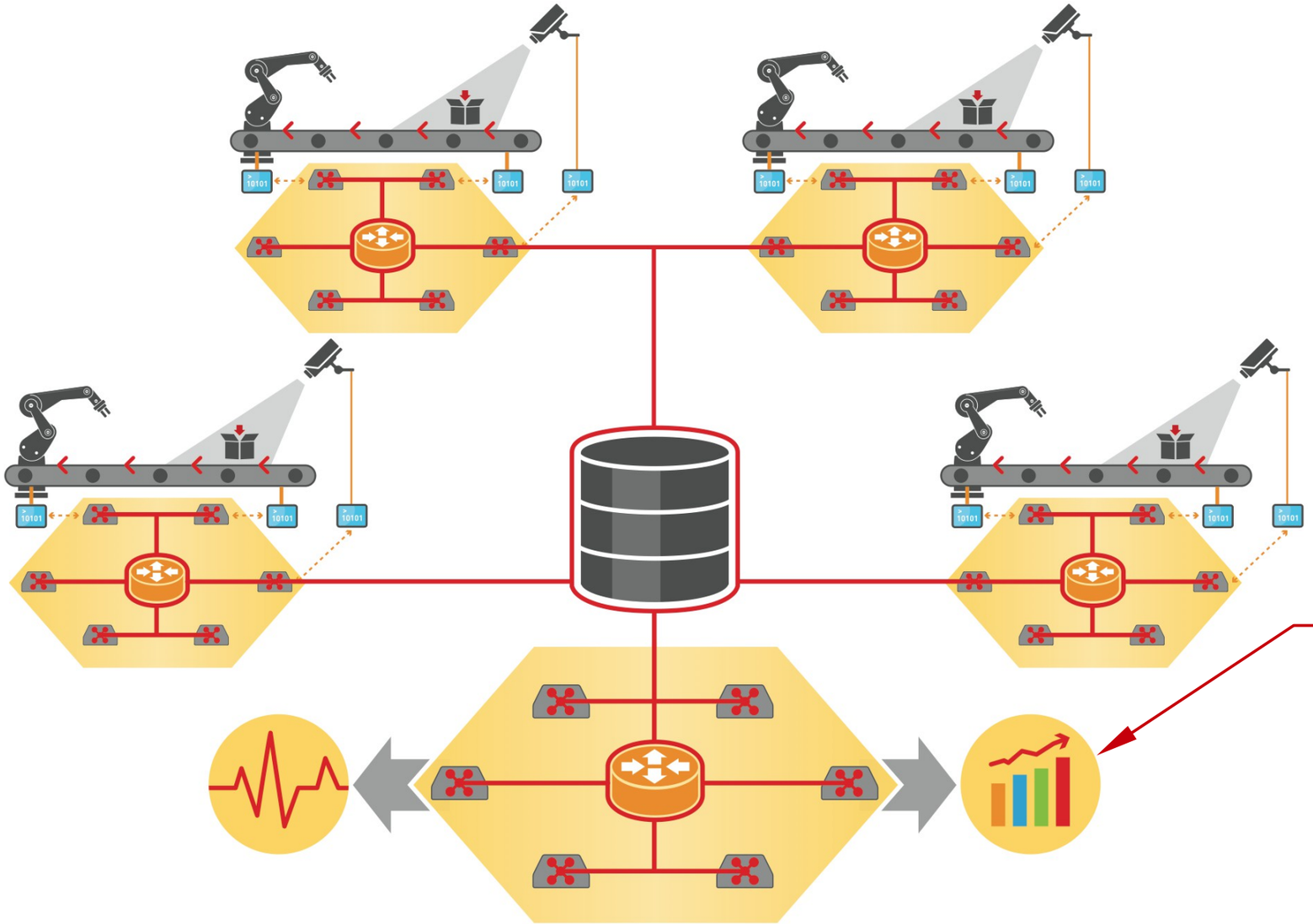


Distributed Realtime Application Monitoring

Utilization of industry standards for online tracking of availability of realtime (sub)systems as well as alerting on failures of soft- and hardware components

Distributed Realtime Systems Framework

Company-wide big data analysis



Big data analysis

Derive key performance indicators to track the production quality and to optimize the manufacturing process

Distributed Realtime Systems Framework

Framework for sustainable distributed realtime systems

Almost deterministic runtime behavior	Platform independent implementation	Low cost for spare parts warehousing	Low development costs	Large pool of appropriate developers	High level of adaptability to evolving IT industry	High degree of hardware abstraction

Distributed Realtime Messaging Framework

Distributed Realtime Application Framework

Distributed Realtime Application Web-interface

Distributed Realtime Application Monitoring

Bigger part of today's realtime systems

Deterministic runtime behavior	Platform specific implementation	High cost for spare parts warehousing	High development costs	Rare developers with in-depth system knowledge	Low level of adaptability to evolving IT industry	Low degree of hardware abstraction