SmartPick SDK

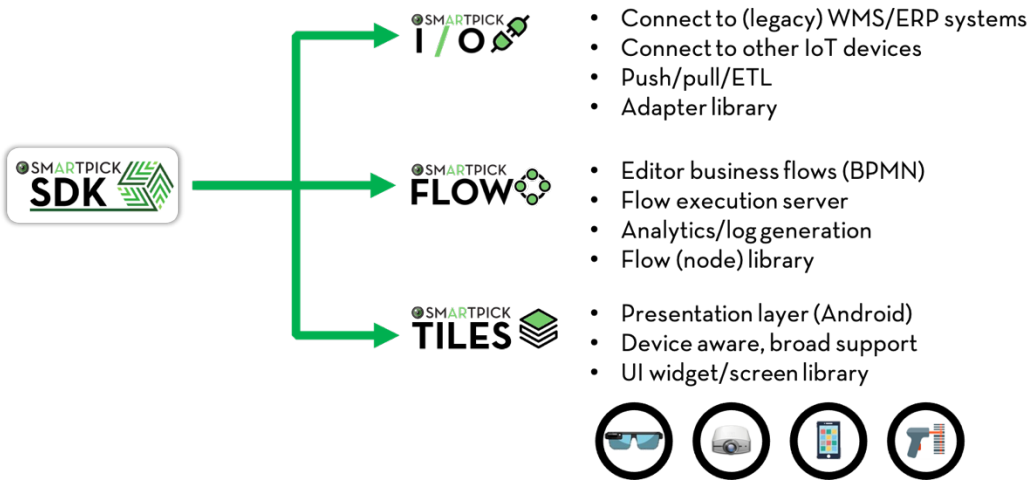
A software platform for

multi-sensory operator guidance

Disclaimer: The information shared in this whitepaper is not meant to be comprehensive or intend to create or imply a contractual relationship regarding the system described. The document is provided for high-level information purposes only.

1 Introduction

SmartPick SDK is a software platform that enables true multi-sensory operator guidance in a logistics/operations context. The business goal is to elegantly connect to any existing IT system, create and optimize business flows with a focus on operations without technical details, and allow the operator to interact with the system and the environment by using the right device(s) for the right task at the right time.
The key focus is on Visual and Augmented reality support, in combination with other suitable hands-free devices and installations. Depending on the use case the device that presents the information to the end-user can be Smart glasses, projectors, tablets (for example mounted on a forklift), handheld scanners, wrist computers or Smart watches. Technically it is possible to work with any type of Android based device.



Through clean separation of concerns and efficient implementation, the platform consists of:

1. **SmartPick I/O**: Universal data connector to any back-end system, ranging from legacy terminal-based systems to contemporary API formats

2. **SmartPick FLOW**: Direct editing and execution of device-agnostic business process flows, using the standardized BPMN notation and execution engine

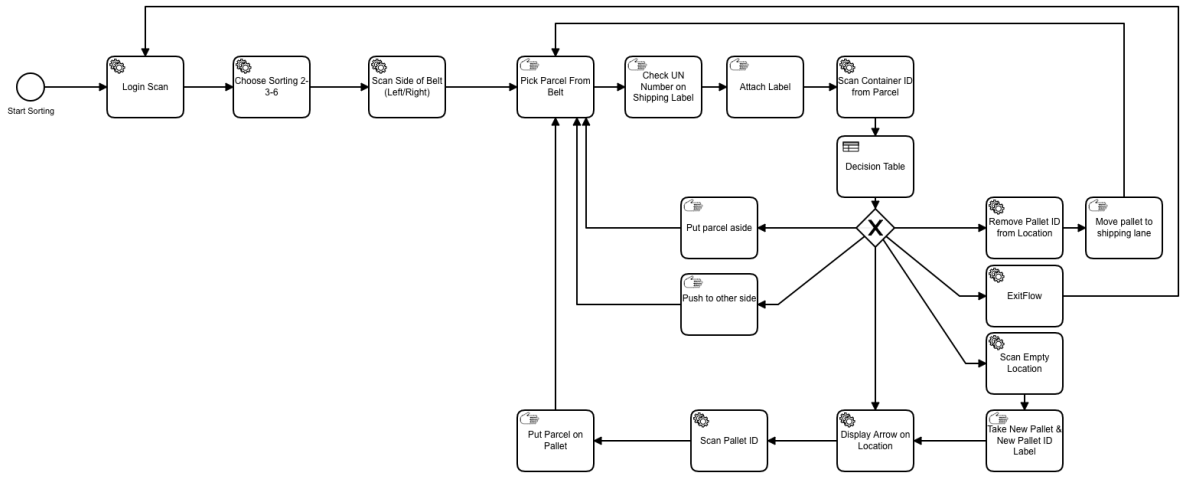
3. **SmartPick TILES**: Flexible presentation layer on Android for multi-device interaction with systems and operators supporting a range of Smart glasses (for Augmented Reality, routing and guidance, ...), tablets, projectors, scanners, wrist computers, etc.

2 SmartPick FLOW: Create and execute business flows

SmartPick FLOW allows you to edit device-agnostic business process flows, using the standardized BPMN notation. Next to that, FLOW also serves as the execution engine during operations. In other words: what you design is actually executed, there is no coding involved to transform the design into a working flow.

Why BPMN ? BPMN Provides an intuitive and easy way for non-expert users to understand and create a process diagram. It can represent the semantics of complex processes easily and in an intelligible form and thus reduce noise in the communication between the process design stage and implementation, execution and management afterwards. This significantly facilitates the integration of other professionals involved in the design process such as business analysts and staff performing the actual processes.

Standard logistics and operations flows can be efficiently expressed with BPMN, through the range of suitable nodes/components and a simple network drawing style. The example below shows a simple BPMN sorting flow.



Components SmartPick FLOW uses the Camunda Workflow engine (https://camunda.com/), an open-source and established BPMN engine and modeler. Under the hood, the different business process tasks are associated with software classes (using the Java programming language) which handle the communication with both SmartPick I/O (for any data connections) as with SmartPick TILES (presentation layer). The communication with the SmartPick I/O layer is done through the ActiveMQ message protocol

(https://activemq.apache.org/), while the communication with the operator devices uses MQTT (https://mqtt.org/), the standard for IoT messaging.
During execution of business flows, SmartPick's event-based design can trigger communications with both back-end data systems and end-user presentation components, to fetch new data, perform database checks, interact with other IoT systems, etc.

Why MQTT? MQTT is the go-to choice for IoT communication. Clients are light-weight and are well suited to run on mobile devices. Meanwhile, the protocol allows reliable, bi-directional communication at scale, despite environments of possibly poor network quality or connectivity. Supporting industry-strengt security and encryption, MQTT allows connections to other IoT enabled services or installations in industrial environments.

2.1 Hands-on with FLOW

To use the FLOW module in your setup, SmartPick provides a basic set of standard flows such as different flavors of picking, replenishment and sorting flows. These can be used as a starting point for further refinement to suit specific needs.

In case the existing library of components does not cover all the needs, additional nodes can be created by means of a provided FLOW API, to allow Java developers to add functionality above the BPMN flows, which can then be executed inside SmartPick in conjunction with existing components.

During execution, FLOW can collect/present extensive and customizable logging for further analysis. This is the first step in iterative process optimization or internal analytics or dashboarding.

3 SmartPick TILES: Presentation and Interaction

SmartPick TILES interacts with the end-users, possibly using multiple devices and sensory in/outputs. TILES translates BPMN business actions from FLOW into device presentations and interactions, tuned to the capabilities of the specific device.

The end-user devices (Smart Glasses, Projectors, Tablets, Wrist computers, ..) contain no specific business logic. The devices accept MQTT Messages from the SmartPick FLOW server and react accordingly. This can be by interacting with the operator's device (e.g. updating instructions in Augmented reality) and possibly communicating back with the server to provide data after scanning, swiping, image recognition etc.

3.1 TILES as a UI component system

TILES comprises a component-based UI system, similar to other 'card' based systems. Individual TILEs represent units of visualization or interaction, that can be combined to build a suitable interface experience. The composition of such interfaces is again device-agnostic and will be converted to the suitable interface by the TILES software during execution. This system of configurable component composition allows maximum flexibility and re-use, hence boosts efficiency for development and later change execution.

3.2 TILES Library of components

A tile can have different layouts and can mix images, text and other representations.
Layouts define how the information is displayed. Some examples of typical layouts are:

- Alert: An alert with a large, centered icon and a message and footnote
- Caption: Images appear full screen in the background with a text caption
- Columns: A two-column layout with centered text on the top and centered image below. The size of the text is dynamic based on the amount of content in the tile.
- Columns Fixed: A two-column layout with centered text on the top and centered image below. The size of the text is dynamic based on the amount of content in the tile
- Rows: A three-row layout with a centered text on first on last row and a centered image in the middle

- Embed: Allows a custom layout to be embedded inside a tile. E.g to display a countdown clock or a camera stream for scanning.
- Text: Text that fills the whole Tile, with optional background images. The size of the text is dynamic based on the amount of content in the Tile.
- Text-Fixed: Text that fills the whole Tile, with optional background images. The size of the text is fixed.
- Camera: An Embed tile designed to use the camera stream. This tile can be used for OCR or AR purposes.
- Scanner: An Embed tile with a surface object to stream the camera input in to decode the barcodes in the camera stream. Different SDK's can be used behind this Scanner tile (Manatee Works, Scandit, Honeywell Swift Decoder, ZXCrossing...)
- Tabbed Text and Grids
- HTML content

The different layout types are defined by configuration (in an XML format) and displayed depending on the device executing the TILES code. For example, a Column layout can have a different look & feel on a Smart Glass then on a Tablet or a Projector.

3.3 Hands-on with TILES

Step 1: Combine tiles: Using the flexibility of TILES, the main area of configuration is the composition/combination of pre-defined TILE components into a UI screens or interactions. This can be done with little to no programming, based on the available elements and configurations. Most of the standard interfaces can be built this way. These composed interactions then have to be linked to commands on the FLOW level, again by configuration.

Step 2: Custom tiles: A next level of configuration can be done by implementing or changing the actual implementation of a TILE for specific devices. A TILES API is provided to allow trained staff to implement and link new TILES by pure coding, unleashing the full power of the device and its interactions. Once implemented, custom TILEs can use the SmartPick stack as any other TILE, with maximum efficiency and re-use.

Step 3: Custom device logic: If very specific business logic is required on the device, TILES allows this to be inserted on a code-level, as long as the basic API constraints of the SmartPick stack as followed. This allows deep integration with client-specific environments by custom implementation.

4 SmartPick I/O: Connect to any system

SmartPick I/O provides all data connection and transformation needs of a SmartPick instance. Although a library of connectors is available, installing SmartPick I/O in a client environment is typically done by SmartPick qualified technical staff.

Standard I/O: SmartPick I/O is built as an event-driven architecture and provides all data operations as a result of triggers from either the SmartPick FLOW or SmartPick TILES layers. Data exchange is done over an ActiveMQ message broker, ensuring efficient and safe communication.

SmartPick I/O can handle a range of formats and protocols, such as

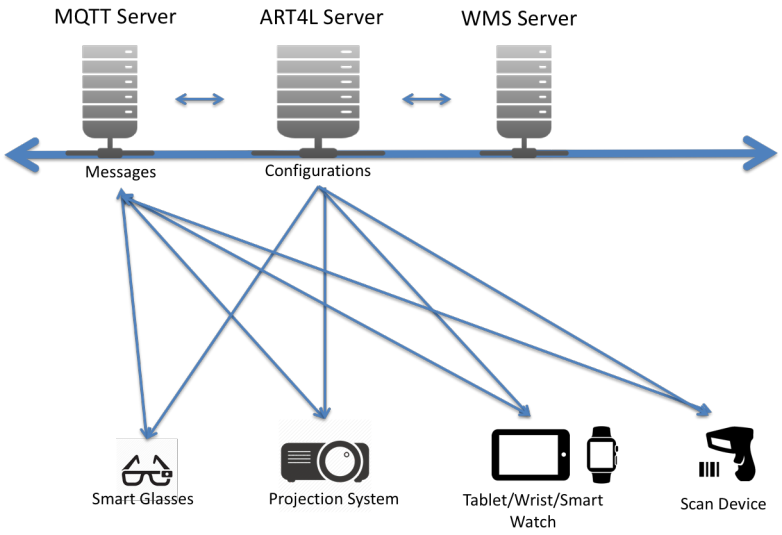
- SOAP with XML data
- REST with JSON data
- FTP file exchange, with parsing of XML, JSON, CSV, XLS, ...
- Raw TCP/IP data exchange
- basic SAP connectors
- Connections with other IoT systems and platforms are also possible

Legacy terminal systems: Next to these, SmartPick I/O can efficiently connect to Telnet (VT100) terminals by means of screen scraping. This solution enables to connect to an AS400 or other telnet-based systems and to read the screens in the background, to harvest the relevant information and simulate a user input by sending keystrokes back. A library is provided to describe and identify screens and their components, allowing efficient and robust implementation with legacy systems without code changes to the back-end system.

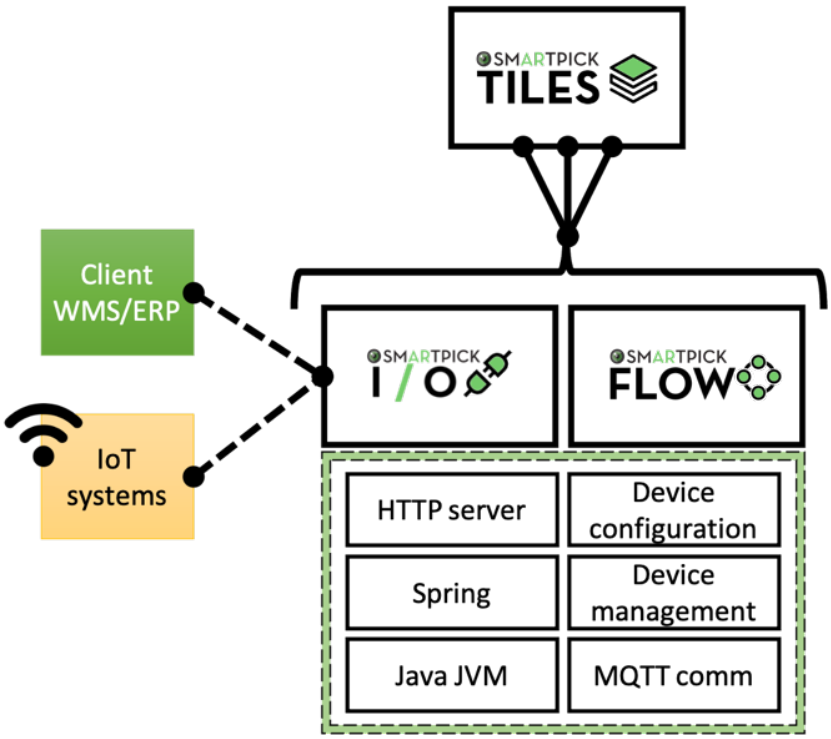
5 SmartPick installations in your environment

The SmartPick solution consists of one or multiple servers implementing a message-based environment. The server(s) provide the SmartPick I/O and SmartPick FLOW components, while the TILES component exists solely on the user devices.

The exchange of commands and data between the components is message based (ActiveMQ and MQTT), except for the initial configuration data needed for setup of the different devices, which is provided by FTP/HTTP. The architecture is designed to support automation of large-scale logistics processes with multiple end-user hands-free devices.



The SmartPick servers are fully Java based and combine different components to implement the full solution, as illustrated in the components diagram. The system is available as a SAAS solution, running as a Kubernetes managed instance on the Google Cloud Platform, or can be installed on premise in single or multi-node redundant setup.


SMARTPICK TILES
Client WMS/ERP
IoT systems
SMARTPICK I/O
SMARTPICK FLOW
HTTP server
Device configuration
Spring
Device management
Java JVM
MQTT comm