All participants:
As the system is a collaborative system, communication between every user is necessary.
Every device (including PC, HTC Vive, Samsung VR, Google Tango) is connected via voice-chat using an existing specification like WebRTC.

Our special features for each of the device are as following:

Personal Computer:

The PC represents the admin of the system. He can switch between all participant's views. He can assign tasks to every participant and change their permissions of interacting with the scene. Tasks consist of multiple orders, e.g. building a stack of multiple blocks. Permissions include the ability of moving, highlighting (including annotating) and marking objects as immutable or invisible. The PC can establish a direct voice-chat to each participant, allowing them to answer vice versa. Recording of the scene, including all movements of objects and participants, is possible. The admin can set constraints to the baxter-robot, e.g. disallowing of specific movements.

HTC Vive

The wearer of the Vive sees the scene as a virtual environment. The user can freely move inside the virtual environment using the tracking of the Vive. The Vive's controllers represent different tools to interact with the scene. These tools allow you to:

- pick and place objects in the scene
- inspect objects to show details
- annotate: mark objects and add notes to them
- zoom-in and -out the scene
- point to objects like a laser-pointer
- communicate, to interact using the voice chat
- warp to a specific location of the scene

Samsung Gear VR

Interacting with the scene is accomplished using a bluetooth paired Dualshock game-controller. The buttons are mapped to the same tools as the ones in the Vive-section of this document. The controllers analog sticks allow the user to move in the scene, head movements are calculated using the phones gyroscope.

Google Tango

Because Google Tango is a tablet, different approaches for interaction with the scene are needed. Most of the tools available for Samsung Gear VR and HTC Vive are also choose-able on the tablet, but controlled differently. E.g. the zoom function is realized using pinch-to-zoom gestures, pointing uses a single tip on the touchscreen and so on. Special views for inspecting objects exist, allowing you to freely rotate and zoom-in and -out the inspected object, as well as getting more detailed information about it.

Implementation:
We plan to implement most of the software using C#. The PC runs the unity-engine and acts as a server using TECS for Publish-Subscribe-Services. Coordinates of non-static objects are extracted

in real-time and published on a certain channel. All clients subscribe to that channel and receive real-time coordinates of these object. Because all clients can also interact with the scene every device also acts as a publisher and therefore has a publish channel as well. Since the communication-API will be realized as Thrift-model, it is possible to translate the API to different programming languages, if needed.