5EHIF HU8 : Smartphone Produktion

Ein asiatisches Fertigungsunternehmen produziert Smartphones für westliche Unternehmen. Es verfügt über drei *Fließbänder* an denen die Smartphones zusammengeschraubt werden.

Auf jedem Band benötigen die Mitarbeiter **zwischen 1 und 3 Sekunden** für die Fertigung **eines** Smartphones.

Jedes Smartphone bekommt bei der Produktion eine eindeutige Nummer zugewiesen - einen sogenannten *universally unique identifier (UUID)*[1].

Hat ein *Fließband* **5 Smartphones** produziert, informiert es die *Lagerabteilung*. Die Lagerabteilung schickt dann einen *Lagerarbeiter* zum Fließband, um die **5 Smartphones** abzuholen. Der Lagerarbeiter braucht seinerseits **2 bis 5 Sekunden**, um zum *Fließband* zu gehen und die Smartphones abzuholen.

Hat das Lager **20 Smartphones** beisammen, informiert es die *Verkaufsabteilung*, dass die Smartphones nun ausgeliefert werden können. Die *Verkaufsabteilung* holt die Smartphones unverzüglich ab und macht sich einen *Log-Eintrag* (in einer `Java:HashMap - C# Dictionary`), der als *key* den aktuellen Zeitstempel [2] und als *value* die 20 Smartphones hat.

Gib jedes Mal, wenn die *Verkaufsabteilung* einen neuen Log-Eintrag macht, alle Log-Einträge mit Zeitstempel und den darin vorhandenen Smartphones aus.

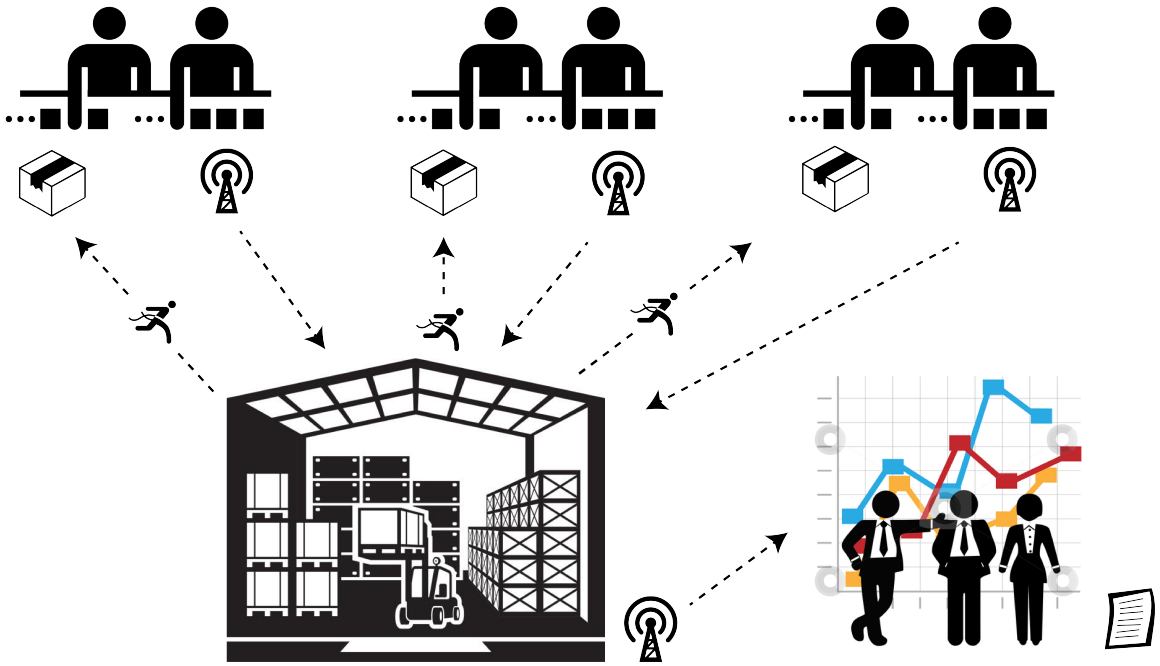

Abbildung 1: Ablauf

[1] Java: Verwende zur Erstellung dieser Nummer die Methode `randomUUID()` der Klasse `UUID` im package `java.util` - C# Beachte, bei MS wird die UUID GUID genannt. Die statische Klasse `Guid` mit Generator `NewGuid()` kann verwendet werden. - Aussehen: `0f8fad5b-d9cb-469f-a165-70867728950e`

[2] Java Sie können hierzu die Klasse `LocalDateTime` aus dem package `java.time` verwenden - C# `DateTime`

Happy coding!

Implementierungsdetails

Die Fließbänder sollen voneinander unabhängig produzieren, sprich eigenständige Threads sein.

Die Information über die 5 fertig produzierten Smartphones soll mittels eines Events versendet werden. Das Fließband soll also **nicht** die Klasse `Lagerabteilung` explizit kennen!

Die Lagerabteilung soll nicht dadurch blockiert sein, dass ein Mitarbeiter zum Fließband geht. Mach also kein `Thread.sleep` in der Klasse `Lagerabteilung`!

Die Klasse `Lagerabteilung` soll die Klasse `Verkaufsabteilung` **nicht** explizit kennen! Sie soll vielmehr über ein Event "allen" Subscribern Bescheid geben, dass ein weiterer Karton mit 20 Smartphones zum Verkauf bereit steht. (*Hinweis: in deiner Implementierung interessiert sich nur die Verkaufsabteilung für dieses Event*)

Achte darauf, dass sich der Lagerarbeiter und das Fließband nicht in die Quere kommen! Die beiden arbeiten unabhängig voneinander. Es ist also theoretisch möglich, dass der Lagerarbeiter gerade die fertigen Smartphones entnimmt, während das Fließband ein weiteres Smartphone dem Bestand hinzufügen will. Sichere die betreffende Codestelle dahingehend ab, dass nur **entweder** der Lagerarbeiter Smartphones *entnimmt* **oder** das Fließband ein neues Gerät *hinzufügt*! Für C# könnte das lock statement dafür verwendet werden: (lock statement (C# reference))

– Gutes Gelingen –

Happy coding!