Karlsruher Institut für Technologie

Vorlesung Softwaretechnik I
Übung 3

SWT I – Sommersemester 2019

Walter F. Tichy, Sebastian Weigelt, Tobias Hey, Martin Blersch

IPD Tichy, Fakultät für Informatik

Einschub-Architektur

AUFGABE 1

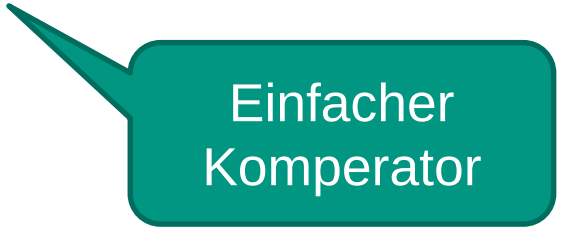
Aufgabe 1 – Plug-In-Architekturen
Plug-Ins ermitteln

```
 public static Iterable<PluginForJmjrst> getPlugins() {
    ServiceLoader<PluginForJmjrst> serviceLoader =
      ServiceLoader.load(PluginForJmjrst.class);

    List<PluginForJmjrst> plugins = new ArrayList<>();
    for (final PluginForJmjrst plugin : serviceLoader) {
      plugins.add(plugin);
    }
    // Sortieren mit dem Comparator aus PluginForJmjrst
    // (siehe nächste Folien)
    Collections.sort(plugins);
    return plugins;
  }
}
```

Aufgabe 1 – Plug-In-Architekturen
Plug-Ins ermitteln

```
public abstract class PluginForJmjrst implements
  Comparable<PluginForJmjrst> {
  // abstrakte Methoden...
  @Override
  public int compareTo(PluginForJmjrst otherPlugin) {
    return Integer.compare( //
        this.getClass().getSimpleName().length(), //
        otherPlugin.getClass().getSimpleName().length() //
    );
  }
}
```



Aufgabe 1 – Plug-In-Architekturen
Menü erweitern

- Anpassen des Konstruktors von Menu in `org.jis.view.Menu`

```
public Menu(Main m) {
  // ...
  option.add(optionen_look);
  option.add(this.set_quality);

  // Plugins fuer JMJRST
  JMenu startPlugins = new
    JMenu(m.mes.getString("Plugins.Start"));
  JMenu configurePlugins = new
    JMenu(m.mes.getString("Plugins.Configure"));
  this.createPluginMenu(m, startPlugins, configurePlugins);
  option.addSeparator();
  option.add(startPlugins);
  option.add(configurePlugins);
  option.addSeparator();
}
```

Neue Menüs erstellen…

… sowie die Menü-Inhalte (Details folgen)

Als Untermenü von “Options” definieren…

… und Trenner hinzufügen

Aufgabe 1 – Plug-In-Architekturen
Menü erweitern

```
private void createPluginMenu(Main m, JMenu sMenu, JMenu cfMenu) {

  for (PluginForJmjrst plugin : PluginManagement.getPlugins()) {
    plugin.init(m);

    JMenuItem start = new JMenuItem(plugin.getName());
    // Seit Java 8 nur noch eine Zeile statt sechs ☺
    start.addActionListener(e -> plugin.run());
    sMenu.add(start);

    if (!plugin.isConfigurable()) {
      continue;
    }

    JMenuItem conf = new JMenuItem(plugin.getName());
    conf.addActionListener(e -> plugin.configure());
    cfMenu.add(conf);
  }
//...
}
```

“Instagrim”

Plug-In zum Menü hinzufügen

Nichts zu konfigurieren?
Dann weiter!

Aufgabe 1 – Plug-In-Architekturen
Menü erweitern

```
private void createPluginMenu(Main m, JMenu sMenu, JMenu cfMenu) {
  //...
  if (startMenu.getMenuComponentCount() == 0) {
    // No Plugins ..
    JMenuItem none =
      new JMenuItem(m.mes.getString("Plugins.None"));
    none.setEnabled(false);
    startMenu.add(none);
  }

  if (configureMenu.getMenuComponentCount() == 0) {
    // No Plugins ..
    JMenuItem none =
      new JMenuItem(m.mes.getString("Plugins.None"));
    none.setEnabled(false);
    configureMenu.add(none);
  }
}
```

Hübsche Ausgabe, falls keine Plugins vorhanden sind (oder nichts zu konfigurieren ist)

War nicht gefordert!

Instagrim-Einschub für iMage

AUFGABE 2

Aufgabe 2 – Instagrim-Einschub

```
@MetaInfServices(PluginForJmjrst.class)
public class InstagrimPlugin extends PluginForJmjrst {

  private Main main;

  @Override
  public boolean isConfigurable
    return true;
  }
  @Override
  public String getName() {
    return “Instagrim";
  }

  //...
}
```

Diese Annotation führt dazu, dass die Textdatei `META-INF/services/org.iMage.plugins.PluginForJmjrst` erzeugt wird.

Sie enthält eine Liste der zugehörigen ServiceProvider, d.h. Plug-Ins.

In unserem Fall: `org.iMage.plugins.instagrim.InstagrimPlugin`

Aufgabe 2 – Instagrim-Einschub

```
@MetaInfServices(PluginForJmjrst.class)
public class InstagrimPlugin extends PluginForJmjrst {
  // ...
  private static final List<String> comments = List.of(
    "iMage - It's simply the best!",
    "Das Bild ist schon nicht mehr von dieser Welt",
    "...");
  private Random rand = new Random();

  // ...
  @Override
  public void run() {
    System.out.println(comments.get(rand.nextInt(comments.size())));
  }

  @Override
  public void configure() {
    JOptionPane.showMessageDialog(this.main,
        "All Comments:\n\n" + String.join("\n", comments),
        "iMage: Der Bildverschönerer, dem Influencer vertrauen!",
        JOptionPane.INFORMATION_MESSAGE);
  }
  // ...
}
```

ImmutableList (seit Java 9)

Ausführung: Kommentare zufällig ausgeben

Konfiguration: Dialog-Fenster mit Liste aller Kommentare

Aufgabe 2 – Instagrim-Einschub

```
@MetaInfServices(PluginForJmjrst.class)
public class InstagrimPlugin extends PluginForJmjrst {
  // ...
  @Override
  public void init(Main main) {
    this.main = main;
    System.err.println(
      "iMage: Der Bildverschönerer, dem Influencer vertrauen! "
        + "Jetzt bist auch Du Teil unseres Teams, “
        + System.getProperty("user.name"));
  }
}
```

Nutzernamen ausgeben

iMage-Bundle

AUFGABE 3

Aufgabe 3 – Bündelung
Main-Methode

```
public class App {
	public static void main(String[] args) {
		org.jis.Main.main(args);
	}
}
```

Aufgabe 3 – Bündelung
Bibliotheken im Manifest eintragen lassen

```
[...]
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-jar-plugin</artifactId>
  [...]
  <configuration>
    <archive>
      <manifest>
        <addClasspath>true</addClasspath>
        <classpathPrefix>lib/</classpathPrefix>
        <classpathLayoutType>
          repository
        </classpathLayoutType>
        <useUniqueVersions>false</useUniqueVersions>
        <mainClass>org.iMage.bundle.App</mainClass>
       </manifest>
    </archive>
  </configuration>
[...]
```

Vermeidung von Namens-Konflikten

Hinzufügen der Hauptklasse in das Manifest (siehe Blatt 2)

Aufgabe 3 – Bündelung
Abhängigkeiten kopieren

```
[...]
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-dependency-plugin</artifactId>
  [...]
    <executions>
      <execution>
        <id>copy-dependencies</id>
        <phase>package</phase>
        <goals>
          <goal>copy-dependencies</goal>
        </goals>
        <configuration>
          <useRepositoryLayout>true</useRepositoryLayout>
          [...]
          <overWriteSnapshots>true</overWriteSnapshots>
          <overWriteIfNewer>true</overWriteIfNewer>
          <outputDirectory>${project.build.directory}/lib</outputDirectory>
          <includeScope>runtime</includeScope>
        </configuration>
      </execution>
    </executions>
  [...]
</plugin>
[...]
```

Beim Aufruf von **`mvn package`** werden Abhängigkeiten kopiert

Aufgabe 3 – Bündelung
Abhängigkeiten kopieren

```
[...]
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-dependency-plugin</artifactId>
  [...]
    <executions>
      <execution>
        <id>copy-dependencies</id>
        <phase>package</phase>
        <goals>
          <goal>copy-dependencies</goal>
        </goals>
        <configuration>
          <useRepositoryLayout>true</useRepositoryLayout>
          [...]
          <overWriteSnapshots>true</overWriteSnapshots>
          <overWriteIfNewer>true</overWriteIfNewer>
          <outputDirectory>${project.build.directory}/lib</outputDirectory>
          <includeScope>runtime</includeScope>
        </configuration>
      </execution>
    </executions>
  [...]
</plugin>
[...]
```

Übernehme Depot-Ordner-Struktur, z.B.: …/target/lib/junit/junit/4.12/junit-4.12.jar

Aufgabe 3 – Bündelung
Abhängigkeiten kopieren

```
[...]
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-dependency-plugin</artifactId>
  [...]
    <executions>
      <execution>
        <id>copy-dependencies</id>
        <phase>package</phase>
        <goals>
          <goal>copy-dependencies</goal>
        </goals>
        <configuration>
          <useRepositoryLayout>true</useRepositoryLayout>
          [...]
          <overWriteSnapshots>true</overWriteSnapshots>
          <overWriteIfNewer>true</overWriteIfNewer>
          <outputDirectory>${project.build.directory}/lib</outputDirectory>
          <includeScope>runtime</includeScope>
        </configuration>
      </execution>
    </executions>
  [...]
</plugin>
[...]
```

Immer die aktuelle Version der Abhängigkeit kopieren

Aufgabe 3 – Bündelung
Abhängigkeiten kopieren

```
[...]
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-dependency-plugin</artifactId>
  [...]
    <executions>
      <execution>
        <id>copy-dependencies</id>
        <phase>package</phase>
        <goals>
          <goal>copy-dependencies</goal>
        </goals>
        <configuration>
          <useRepositoryLayout>true</useRepositoryLayout>
          [...]
          <overWriteSnapshots>true</overWriteSnapshots>
          <overWriteIfNewer>true</overWriteIfNewer>
          <outputDirectory>${project.build.directory}/lib</outputDirectory>
          <includeScope>runtime</includeScope>
        </configuration>
      </execution>
    </executions>
  [...]
</plugin>
[...]
```

Zielverzeichnis: `/lib` (vgl. jar-plugin → classpathPrefix)

Aufgabe 3 – Bündelung
Abhängigkeiten kopieren

```
[...]
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-dependency-plugin</artifactId>
  [...]
    <executions>
      <execution>
        <id>copy-dependencies</id>
        <phase>package</phase>
        <goals>
          <goal>copy-dependencies</goal>
        </goals>
        <configuration>
          <useRepositoryLayout>true</useRepositoryLayout>
          [...]
          <overWriteSnapshots>true</overWriteSnapshots>
          <overWriteIfNewer>true</overWriteIfNewer>
          <outputDirectory>${project.build.directory}/lib</outputDirectory>
          <includeScope>runtime</includeScope>
        </configuration>
      </execution>
    </executions>
  [...]
</plugin>
[...]
```

kopiere Abhängigkeiten mit `scope runtime` oder `compile`

also junit schon mal nicht ☺

UML-Aktivitätsdiagramm

AUFGABE 4

Aufgabe 4 - Aktivitätsdiagramm

Zu Beginn zeigt die Applikation die Startseite an. Der Nutzer kann sich per Klick auf ein HDR-Bild von der Applikation eine Vorschau erzeugen und anzeigen lassen. Daraufhin betätigt der Nutzer die „Teilen“-Schaltfläche, um das Bild auf Facezine hochzuladen. Dies sorgt dafür, dass das entsprechende Bild auf dem Pear-Corp-Zentralserver gesucht und von dort an Facezine gesendet wird. Gleichzeitig mit dem Senden kehrt die Applikation auf die Startseite zurück. Über das Klicken der Schaltfläche „Neues HDR-Bild“ gelangt der Nutzer zu einer Übersicht der auf dem Mobiltelefon gespeicherten Bilder. Der Nutzer wählt drei Bilder einer Bildreihe aus und bestätigt seine Auswahl mit einem Klick auf „HDR-Bild erstellen“. Die Eingabebilder werden anschließend auf den Pear-Corp-Zentralserver hochgeladen und das HDR-Bild erzeugt. Zur Erzeugung des HDR-Bildes rekonstruiert der Server zunächst die Antwortkurve der Kamera der Eingabebilder. Daraufhin kombiniert er die Eingabebilder zu einem HDR-Bild. Hierzu iteriert der Server über die Pixel der Bilder und bestimmt für jeden Pixel den neuen Farbwert. Dazu kombiniert er zunächst die gewichteten Einzelfarbwerte mit der Antwortkurve. Anschließend wird über die Summe der Gewichte normiert. Wurde über alle Pixel iteriert, bildet der Server das Bild in den RGB-Farbraum ab und speichert sowohl das HDR- als auch das RGB-Bild auf dem Server. Die Applikation wartet, solange die Erzeugung nicht beendet ist. Per Klick auf die Schaltfläche „HDR-Bild abholen“ wird dem Nutzer das Vorschaubild und Kaufoptionen angezeigt. Hat der Nutzer eine Kaufoption gewählt, speichert die Applikation das Bild im Bilderordner des Mobiltelefons. Besitzt der Nutzer bereits ein Monatsabonnement, sollen keine Kaufoptionen, sondern direkt eine Schaltfläche „HDR-Bild jetzt speichern“ angezeigt werden.

Aufgabe 4 - Aktivitätsdiagramm

Zu Beginn zeigt die Applikation die Startseite an. Der Nutzer kann sich per Klick auf ein HDR-Bild von der Applikation eine Vorschau erzeugen und anzeigen lassen. Daraufhin betätigt der Nutzer die „Teilen“-Schaltfläche, um das Bild auf Facezine hochzuladen. Dies sorgt dafür, dass das entsprechende Bild auf dem Pear-Corp-Zentralserver gesucht und von dort an Facezine gesendet wird. Gleichzeitig mit dem Senden kehrt die Applikation auf die Startseite zurück. Über das Klicken der Schaltfläche „Neues HDR-Bild“ gelangt der Nutzer zu einer Übersicht der auf dem Mobiltelefon gespeicherten Bilder. Der Nutzer wählt drei Bilder einer Bildreihe aus und bestätigt seine Auswahl mit einem Klick auf „HDR-Bild erstellen“. Die Eingabebilder werden anschließend auf den Pear-Corp-Zentralserver hochgeladen und das HDR-Bild erzeugt. Zur Erzeugung des HDR-Bildes rekonstruiert der Server zunächst die Antwortkurve der Kamera der Eingabebilder. Daraufhin kombiniert er die Eingabebilder zu einem HDR-Bild. Hierzu iteriert der Server über die Pixel der Bilder und bestimmt für jeden Pixel den neuen Farbwert. Dazu kombiniert er zunächst die gewichteten Einzelfarbwerte mit der Antwortkurve. Anschließend wird über die Summe der Gewichte normiert. Wurde über alle Pixel iteriert, bildet der Server das Bild in den RGB-Farbraum ab und speichert sowohl das HDR- als auch das RGB-Bild auf dem Server. Die Applikation wartet, solange die Erzeugung nicht beendet ist. Per Klick auf die Schaltfläche „HDR-Bild abholen“ wird dem Nutzer das Vorschaubild und Kaufoptionen angezeigt. Hat der Nutzer eine Kaufoption gewählt, speichert die Applikation das Bild im Bilderordner des Mobiltelefons. Besitzt der Nutzer bereits ein Monatsabonnement, sollen keine Kaufoptionen, sondern direkt eine Schaltfläche „HDR-Bild jetzt speichern“ angezeigt werden.

Aufgabe 4 - Aktivitätsdiagramm

Zu Beginn zeigt die Applikation die Startseite an. Der Nutzer kann sich per Klick auf ein HDR-Bild von der Applikation eine Vorschau erzeugen und anzeigen lassen. Daraufhin betätigt der Nutzer die „Teilen“-Schaltfläche, um das Bild auf Facezine hochzuladen. Dies sorgt dafür, dass das entsprechende Bild auf dem Pear-Corp-Zentralserver gesucht und von dort an Facezine gesendet wird. Gleichzeitig mit dem Senden kehrt die Applikation auf die Startseite zurück. Über das Klicken der Schaltfläche „Neues HDR-Bild“ gelangt der Nutzer zu einer Übersicht der auf dem Mobiltelefon gespeicherten Bilder. Der Nutzer wählt drei Bilder einer Bildreihe aus und bestätigt seine Auswahl mit einem Klick auf „HDR-Bild erstellen“. Die Eingabebilder werden anschließend auf den Pear-Corp-Zentralserver hochgeladen und das HDR-Bild erzeugt. Zur Erzeugung des HDR-Bildes rekonstruiert der Server zunächst die Antwortkurve der Kamera der Eingabebilder. Daraufhin kombiniert er die Eingabebilder zu einem HDR-Bild. Hierzu iteriert der Server über die Pixel der Bilder und bestimmt für jeden Pixel den neuen Farbwert. Dazu kombiniert er zunächst die gewichteten Einzelfarbwerte mit der Antwortkurve. Anschließend wird über die Summe der Gewichte normiert. Wurde über alle Pixel iteriert, bildet der Server das Bild in den RGB-Farbraum ab und speichert sowohl das HDR- als auch das RGB-Bild auf dem Server. Die Applikation wartet, solange die Erzeugung nicht beendet ist. Per Klick auf die Schaltfläche „HDR-Bild abholen“ wird dem Nutzer das Vorschaubild und Kaufoptionen angezeigt. Hat der Nutzer eine Kaufoption gewählt, speichert die Applikation das Bild im Bilderordner des Mobiltelefons. Besitzt der Nutzer bereits ein Monatsabonnement, sollen keine Kaufoptionen, sondern direkt eine Schaltfläche „HDR-Bild jetzt speichern“ angezeigt werden.

Aufgabe 4 - Aktivitätsdiagramm

Zu Beginn zeigt die Applikation die Startseite an. Der Nutzer kann sich per Klick auf ein HDR-Bild von der Applikation eine Vorschau erzeugen und anzeigen lassen. Daraufhin betätigt der Nutzer die „Teilen“-Schaltfläche, um das Bild auf Facezine hochzuladen. Dies sorgt dafür, dass das entsprechende Bild auf dem Pear-Corp-Zentralserver gesucht und von dort an Facezine gesendet wird. Gleichzeitig mit dem Senden kehrt die Applikation auf die Startseite zurück. Über das Klicken der Schaltfläche „Neues HDR-Bild“ gelangt der Nutzer zu einer Übersicht der auf dem Mobiltelefon gespeicherten Bilder. Der Nutzer wählt drei Bilder einer Bildreihe aus und bestätigt seine Auswahl mit einem Klick auf „HDR-Bild erstellen“. Die Eingabebilder werden anschließend auf den Pear-Corp-Zentralserver hochgeladen und das HDR-Bild erzeugt. Zur Erzeugung des HDR-Bildes rekonstruiert der Server zunächst die Antwortkurve der Kamera der Eingabebilder. Daraufhin kombiniert er die Eingabebilder zu einem HDR-Bild. Hierzu iteriert der Server über die Pixel der Bilder und bestimmt für jeden Pixel den neuen Farbwert. Dazu kombiniert er zunächst die gewichteten Einzelfarbwerte mit der Antwortkurve. Anschließend wird über die Summe der Gewichte normiert. Wurde über alle Pixel iteriert, bildet der Server das Bild in den RGB-Farbraum ab und speichert sowohl das HDR- als auch das RGB-Bild auf dem Server. Die Applikation wartet, solange die Erzeugung nicht beendet ist. Per Klick auf die Schaltfläche „HDR-Bild abholen“ wird dem Nutzer das Vorschaubild und Kaufoptionen angezeigt. Hat der Nutzer eine Kaufoption gewählt, speichert die Applikation das Bild im Bilderordner des Mobiltelefons. Besitzt der Nutzer bereits ein Monatsabonnement, sollen keine Kaufoptionen, sondern direkt eine Schaltfläche „HDR-Bild jetzt speichern“ angezeigt werden.

Aufgabe 4 - Aktivitätsdiagramm

Zu Beginn zeigt die Applikation die Startseite an. Der Nutzer kann sich per Klick auf ein HDR-Bild von der Applikation eine Vorschau erzeugen und anzeigen lassen. Daraufhin betätigt der Nutzer die „Teilen“-Schaltfläche, um das Bild auf Facezine hochzuladen. Dies sorgt dafür, dass das entsprechende Bild auf dem Pear-Corp-Zentralserver gesucht und von dort an Facezine gesendet wird. Gleichzeitig mit dem Senden kehrt die Applikation auf die Startseite zurück. Über das Klicken der Schaltfläche „Neues HDR-Bild“ gelangt der Nutzer zu einer Übersicht der auf dem Mobiltelefon gespeicherten Bilder. Der Nutzer wählt drei Bilder einer Bildreihe aus und bestätigt seine Auswahl mit einem Klick auf „HDR-Bild erstellen“. Die Eingabebilder werden anschließend auf den Pear-Corp-Zentralserver hochgeladen und das HDR-Bild erzeugt. Zur Erzeugung des HDR-Bildes rekonstruiert der Server zunächst die Antwortkurve der Kamera der Eingabebilder. Daraufhin kombiniert er die Eingabebilder zu einem HDR-Bild. Hierzu iteriert der Server über die Pixel der Bilder und bestimmt für jeden Pixel den neuen Farbwert. Dazu kombiniert er zunächst die gewichteten Einzelfarbwerte mit der Antwortkurve. Anschließend wird über die Summe der Gewichte normiert. Wurde über alle Pixel iteriert, bildet der Server das Bild in den RGB-Farbraum ab und speichert sowohl das HDR- als auch das RGB-Bild auf dem Server. Die Applikation wartet, solange die Erzeugung nicht beendet ist. Per Klick auf die Schaltfläche „HDR-Bild abholen“ wird dem Nutzer das Vorschaubild und Kaufoptionen angezeigt. Hat der Nutzer eine Kaufoption gewählt, speichert die Applikation das Bild im Bilderordner des Mobiltelefons. Besitzt der Nutzer bereits ein Monatsabonnement, sollen keine Kaufoptionen, sondern direkt eine Schaltfläche „HDR-Bild jetzt speichern“ angezeigt werden.

Aufgabe 4 - Aktivitätsdiagramm

Zu Beginn zeigt die Applikation die Startseite an. Der Nutzer kann sich per Klick auf ein HDR-Bild von der Applikation eine Vorschau erzeugen und anzeigen lassen. Daraufhin betätigt der Nutzer die „Teilen“-Schaltfläche, um das Bild auf Facezine hochzuladen. Dies sorgt dafür, dass das entsprechende Bild auf dem Pear-Corp-Zentralserver gesucht und von dort an Facezine gesendet wird. Gleichzeitig mit dem Senden kehrt die Applikation auf die Startseite zurück. Über das Klicken der Schaltfläche „Neues HDR-Bild“ gelangt der Nutzer zu einer Übersicht der auf dem Mobiltelefon gespeicherten Bilder. Der Nutzer wählt drei Bilder einer Bildreihe aus und bestätigt seine Auswahl mit einem Klick auf „HDR-Bild erstellen“. Die Eingabebilder werden anschließend auf den Pear-Corp-Zentralserver hochgeladen und das HDR-Bild erzeugt. Zur Erzeugung des HDR-Bildes rekonstruiert der Server zunächst die Antwortkurve der Kamera der Eingabebilder. Daraufhin kombiniert er die Eingabebilder zu einem HDR-Bild. Hierzu iteriert der Server über die Pixel der Bilder und bestimmt für jeden Pixel den neuen Farbwert. Dazu kombiniert er zunächst die gewichteten Einzelfarbwerte mit der Antwortkurve. Anschließend wird über die Summe der Gewichte normiert. Wurde über alle Pixel iteriert, bildet der Server das Bild in den RGB-Farbraum ab und speichert sowohl das HDR- als auch das RGB-Bild auf dem Server. Die Applikation wartet, solange die Erzeugung nicht beendet ist. Per Klick auf die Schaltfläche „HDR-Bild abholen“ wird dem Nutzer das Vorschaubild und Kaufoptionen angezeigt. Hat der Nutzer eine Kaufoption gewählt, speichert die Applikation das Bild im Bilderordner des Mobiltelefons. Besitzt der Nutzer bereits ein Monatsabonnement, sollen keine Kaufoptionen, sondern direkt eine Schaltfläche „HDR-Bild jetzt speichern“ angezeigt werden.

Aufgabe 4 - Aktivitätsdiagramm

Zu Beginn zeigt die Applikation die Startseite an. Der Nutzer kann sich per Klick auf ein HDR-Bild von der Applikation eine Vorschau erzeugen und anzeigen lassen. Daraufhin betätigt der Nutzer die „Teilen“-Schaltfläche, um das Bild auf Facezine hochzuladen. Dies sorgt dafür, dass das entsprechende Bild auf dem Pear-Corp-Zentralserver gesucht und von dort an Facezine gesendet wird. Gleichzeitig mit dem Senden kehrt die Applikation auf die Startseite zurück. Über das Klicken der Schaltfläche „Neues HDR-Bild“ gelangt der Nutzer zu einer Übersicht der auf dem Mobiltelefon gespeicherten Bilder. Der Nutzer wählt drei Bilder einer Bildreihe aus und bestätigt seine Auswahl mit einem Klick auf „HDR-Bild erstellen“. Die Eingabebilder werden anschließend auf den Pear-Corp-Zentralserver hochgeladen und das HDR-Bild erzeugt. Zur Erzeugung des HDR-Bildes rekonstruiert der Server zunächst die Antwortkurve der Kamera der Eingabebilder. Daraufhin kombiniert er die Eingabebilder zu einem HDR-Bild. Hierzu iteriert der Server über die Pixel der Bilder und bestimmt für jeden Pixel den neuen Farbwert. Dazu kombiniert er zunächst die gewichteten Einzelfarbwerte mit der Antwortkurve. Anschließend wird über die Summe der Gewichte normiert. Wurde über alle Pixel iteriert, bildet der Server das Bild in den RGB-Farbraum ab und speichert sowohl das HDR- als auch das RGB-Bild auf dem Server. Die Applikation wartet, solange die Erzeugung nicht beendet ist. Per Klick auf die Schaltfläche „HDR-Bild abholen“ wird dem Nutzer das Vorschaubild und Kaufoptionen angezeigt. Hat der Nutzer eine Kaufoption gewählt, speichert die Applikation das Bild im Bilderordner des Mobiltelefons. Besitzt der Nutzer bereits ein Monatsabonnement, sollen keine Kaufoptionen, sondern direkt eine Schaltfläche „HDR-Bild jetzt speichern“ angezeigt werden.

Aufgabe 4 - Aktivitätsdiagramm

Zu Beginn zeigt die Applikation die Startseite an. Der Nutzer kann sich per Klick auf ein HDR-Bild von der Applikation eine Vorschau erzeugen und anzeigen lassen. Daraufhin betätigt der Nutzer die „Teilen“-Schaltfläche, um das Bild auf Facezine hochzuladen. Dies sorgt dafür, dass das entsprechende Bild auf dem Pear-Corp-Zentralserver gesucht und von dort an Facezine gesendet wird. Gleichzeitig mit dem Senden kehrt die Applikation auf die Startseite zurück. Über das Klicken der Schaltfläche „Neues HDR-Bild“ gelangt der Nutzer zu einer Übersicht der auf dem Mobiltelefon gespeicherten Bilder. Der Nutzer wählt drei Bilder einer Bildreihe aus und bestätigt seine Auswahl mit einem Klick auf „HDR-Bild erstellen“. Die Eingabebilder werden anschließend auf den Pear-Corp-Zentralserver hochgeladen und das HDR-Bild erzeugt. Zur Erzeugung des HDR-Bildes rekonstruiert der Server zunächst die Antwortkurve der Kamera der Eingabebilder. Daraufhin kombiniert er die Eingabebilder zu einem HDR-Bild. Hierzu iteriert der Server über die Pixel der Bilder und bestimmt für jeden Pixel den neuen Farbwert. Dazu kombiniert er zunächst die gewichteten Einzelfarbwerte mit der Antwortkurve. Anschließend wird über die Summe der Gewichte normiert. Wurde über alle Pixel iteriert, bildet der Server das Bild in den RGB-Farbraum ab und speichert sowohl das HDR- als auch das RGB-Bild auf dem Server. Die Applikation wartet, solange die Erzeugung nicht beendet ist. Per Klick auf die Schaltfläche „HDR-Bild abholen“ wird dem Nutzer das Vorschaubild und Kaufoptionen angezeigt. Hat der Nutzer eine Kaufoption gewählt, speichert die Applikation das Bild im Bilderordner des Mobiltelefons. Besitzt der Nutzer bereits ein Monatsabonnement, sollen keine Kaufoptionen, sondern direkt eine Schaltfläche „HDR-Bild jetzt speichern“ angezeigt werden.

Aufgabe 4 - Aktivitätsdiagramm

Zu Beginn zeigt die Applikation die Startseite an. Der Nutzer kann sich per Klick auf ein HDR-Bild von der Applikation eine Vorschau erzeugen und anzeigen lassen. Daraufhin betätigt der Nutzer die „Teilen“-Schaltfläche, um das Bild auf Facezine hochzuladen. Dies sorgt dafür, dass das entsprechende Bild auf dem Pear-Corp-Zentralserver gesucht und von dort an Facezine gesendet wird. Gleichzeitig mit dem Senden kehrt die Applikation auf die Startseite zurück. Über das Klicken der Schaltfläche „Neues HDR-Bild“ gelangt der Nutzer zu einer Übersicht der auf dem Mobiltelefon gespeicherten Bilder. Der Nutzer wählt drei Bilder einer Bildreihe aus und bestätigt seine Auswahl mit einem Klick auf „HDR-Bild erstellen“. Die Eingabebilder werden anschließend auf den Pear-Corp-Zentralserver hochgeladen und das HDR-Bild erzeugt. Zur Erzeugung des HDR-Bildes rekonstruiert der Server zunächst die Antwortkurve der Kamera der Eingabebilder. Daraufhin kombiniert er die Eingabebilder zu einem HDR-Bild. Hierzu iteriert der Server über die Pixel der Bilder und bestimmt für jeden Pixel den neuen Farbwert. Dazu kombiniert er zunächst die gewichteten Einzelfarbwerte mit der Antwortkurve. Anschließend wird über die Summe der Gewichte normiert. Wurde über alle Pixel iteriert, bildet der Server das Bild in den RGB-Farbraum ab und speichert sowohl das HDR- als auch das RGB-Bild auf dem Server. Die Applikation wartet, solange die Erzeugung nicht beendet ist. Per Klick auf die Schaltfläche „HDR-Bild abholen“ wird dem Nutzer das Vorschaubild und Kaufoptionen angezeigt. Hat der Nutzer eine Kaufoption gewählt, speichert die Applikation das Bild im Bilderordner des Mobiltelefons. Besitzt der Nutzer bereits ein Monatsabonnement, sollen keine Kaufoptionen, sondern direkt eine Schaltfläche „HDR-Bild jetzt speichern“ angezeigt werden.

Aufgabe 4 - Aktivitätsdiagramm

Zu Beginn zeigt die Applikation die Startseite an. Der Nutzer kann sich per Klick auf ein HDR-Bild von der Applikation eine Vorschau erzeugen und anzeigen lassen. Daraufhin betätigt der Nutzer die „Teilen“-Schaltfläche, um das Bild auf Facezine hochzuladen. Dies sorgt dafür, dass das entsprechende Bild auf dem Pear-Corp-Zentralserver gesucht und von dort an Facezine gesendet wird. Gleichzeitig mit dem Senden kehrt die Applikation auf die Startseite zurück. Über das Klicken der Schaltfläche „Neues HDR-Bild“ gelangt der Nutzer zu einer Übersicht der auf dem Mobiltelefon gespeicherten Bilder. Der Nutzer wählt drei Bilder einer Bildreihe aus und bestätigt seine Auswahl mit einem Klick auf „HDR-Bild erstellen“. Die Eingabebilder werden anschließend auf den Pear-Corp-Zentralserver hochgeladen und das HDR-Bild erzeugt. Zur Erzeugung des HDR-Bildes rekonstruiert der Server zunächst die Antwortkurve der Kamera der Eingabebilder. Daraufhin kombiniert er die Eingabebilder zu einem HDR-Bild. Hierzu iteriert der Server über die Pixel der Bilder und bestimmt für jeden Pixel den neuen Farbwert. Dazu kombiniert er zunächst die gewichteten Einzelfarbwerte mit der Antwortkurve. Anschließend wird über die Summe der Gewichte normiert. Wurde über alle Pixel iteriert, bildet der Server das Bild in den RGB-Farbraum ab und speichert sowohl das HDR- als auch das RGB-Bild auf dem Server. Die Applikation wartet, solange die Erzeugung nicht beendet ist. Per Klick auf die Schaltfläche „HDR-Bild abholen“ wird dem Nutzer das Vorschaubild und Kaufoptionen angezeigt. Hat der Nutzer eine Kaufoption gewählt, speichert die Applikation das Bild im Bilderordner des Mobiltelefons. Besitzt der Nutzer bereits ein Monatsabonnement, sollen keine Kaufoptionen, sondern direkt eine Schaltfläche „HDR-Bild jetzt speichern“ angezeigt werden.

Aufgabe 4 - Aktivitätsdiagramm

Zu Beginn zeigt die Applikation die Startseite an. Der Nutzer kann sich per Klick auf ein HDR-Bild von der Applikation eine Vorschau erzeugen und anzeigen lassen. Daraufhin betätigt der Nutzer die „Teilen“-Schaltfläche, um das Bild auf Facezine hochzuladen. Dies sorgt dafür, dass das entsprechende Bild auf dem Pear-Corp-Zentralserver gesucht und von dort an Facezine gesendet wird. Gleichzeitig mit dem Senden kehrt die Applikation auf die Startseite zurück. Über das Klicken der Schaltfläche „Neues HDR-Bild“ gelangt der Nutzer zu einer Übersicht der auf dem Mobiltelefon gespeicherten Bilder. Der Nutzer wählt drei Bilder einer Bildreihe aus und bestätigt seine Auswahl mit einem Klick auf „HDR-Bild erstellen“. Die Eingabebilder werden anschließend auf den Pear-Corp-Zentralserver hochgeladen und das HDR-Bild erzeugt. Zur Erzeugung des HDR-Bildes rekonstruiert der Server zunächst die Antwortkurve der Kamera der Eingabebilder. Daraufhin kombiniert er die Eingabebilder zu einem HDR-Bild. Hierzu iteriert der Server über die Pixel der Bilder und bestimmt für jeden Pixel den neuen Farbwert. Dazu kombiniert er zunächst die gewichteten Einzelfarbwerte mit der Antwortkurve. Anschließend wird über die Summe der Gewichte normiert. Wurde über alle Pixel iteriert, bildet der Server das Bild in den RGB-Farbraum ab und speichert sowohl das HDR- als auch das RGB-Bild auf dem Server. Die Applikation wartet, solange die Erzeugung nicht beendet ist. Per Klick auf die Schaltfläche „HDR-Bild abholen“ wird dem Nutzer das Vorschaubild und Kaufoptionen angezeigt. Hat der Nutzer eine Kaufoption gewählt, speichert die Applikation das Bild im Bilderordner des Mobiltelefons. Besitzt der Nutzer bereits ein Monatsabonnement, sollen keine Kaufoptionen, sondern direkt eine Schaltfläche „HDR-Bild jetzt speichern“ angezeigt werden.

Aufgabe 4 - Aktivitätsdiagramm

Zu Beginn zeigt die Applikation die Startseite an. Der Nutzer kann sich per Klick auf ein HDR-Bild von der Applikation eine Vorschau erzeugen und anzeigen lassen. Daraufhin betätigt der Nutzer die „Teilen“-Schaltfläche, um das Bild auf Facezine hochzuladen. Dies sorgt dafür, dass das entsprechende Bild auf dem Pear-Corp-Zentralserver gesucht und von dort an Facezine gesendet wird. Gleichzeitig mit dem Senden kehrt die Applikation auf die Startseite zurück. Über das Klicken der Schaltfläche „Neues HDR-Bild“ gelangt der Nutzer zu einer Übersicht der auf dem Mobiltelefon gespeicherten Bilder. Der Nutzer wählt drei Bilder einer Bildreihe aus und bestätigt seine Auswahl mit einem Klick auf „HDR-Bild erstellen“. Die Eingabebilder werden anschließend auf den Pear-Corp-Zentralserver hochgeladen und das HDR-Bild erzeugt. Zur Erzeugung des HDR-Bildes rekonstruiert der Server zunächst die Antwortkurve der Kamera der Eingabebilder. Daraufhin kombiniert er die Eingabebilder zu einem HDR-Bild. Hierzu iteriert der Server über die Pixel der Bilder und bestimmt für jeden Pixel den neuen Farbwert. Dazu kombiniert er zunächst die gewichteten Einzelfarbwerte mit der Antwortkurve. Anschließend wird über die Summe der Gewichte normiert. Wurde über alle Pixel iteriert, bildet der Server das Bild in den RGB-Farbraum ab und speichert sowohl das HDR- als auch das RGB-Bild auf dem Server. Die Applikation wartet, solange die Erzeugung nicht beendet ist. Per Klick auf die Schaltfläche „HDR-Bild abholen“ wird dem Nutzer das Vorschaubild und Kaufoptionen angezeigt. Hat der Nutzer eine Kaufoption gewählt, speichert die Applikation das Bild im Bilderordner des Mobiltelefons. Besitzt der Nutzer bereits ein Monatsabonnement, sollen keine Kaufoptionen, sondern direkt eine Schaltfläche „HDR-Bild jetzt speichern“ angezeigt werden.

Aufgabe 4 - Aktivitätsdiagramm

Zu Beginn zeigt die Applikation die Startseite an. Der Nutzer kann sich per Klick auf ein HDR-Bild von der Applikation eine Vorschau erzeugen und anzeigen lassen. Daraufhin betätigt der Nutzer die „Teilen“-Schaltfläche, um das Bild auf Facezine hochzuladen. Dies sorgt dafür, dass das entsprechende Bild auf dem Pear-Corp-Zentralserver gesucht und von dort an Facezine gesendet wird. Gleichzeitig mit dem Senden kehrt die Applikation auf die Startseite zurück. Über das Klicken der Schaltfläche „Neues HDR-Bild“ gelangt der Nutzer zu einer Übersicht der auf dem Mobiltelefon gespeicherten Bilder. Der Nutzer wählt drei Bilder einer Bildreihe aus und bestätigt seine Auswahl mit einem Klick auf „HDR-Bild erstellen“. Die Eingabebilder werden anschließend auf den Pear-Corp-Zentralserver hochgeladen und das HDR-Bild erzeugt. Zur Erzeugung des HDR-Bildes rekonstruiert der Server zunächst die Antwortkurve der Kamera der Eingabebilder. Daraufhin kombiniert er die Eingabebilder zu einem HDR-Bild. Hierzu iteriert der Server über die Pixel der Bilder und bestimmt für jeden Pixel den neuen Farbwert. Dazu kombiniert er zunächst die gewichteten Einzelfarbwerte mit der Antwortkurve. Anschließend wird über die Summe der Gewichte normiert. Wurde über alle Pixel iteriert, bildet der Server das Bild in den RGB-Farbraum ab und speichert sowohl das HDR- als auch das RGB-Bild auf dem Server. Die Applikation wartet, solange die Erzeugung nicht beendet ist. Per Klick auf die Schaltfläche „HDR-Bild abholen“ wird dem Nutzer das Vorschaubild und Kaufoptionen angezeigt. Hat der Nutzer eine Kaufoption gewählt, speichert die Applikation das Bild im Bilderordner des Mobiltelefons. Besitzt der Nutzer bereits ein Monatsabonnement, sollen keine Kaufoptionen, sondern direkt eine Schaltfläche „HDR-Bild jetzt speichern“ angezeigt werden.

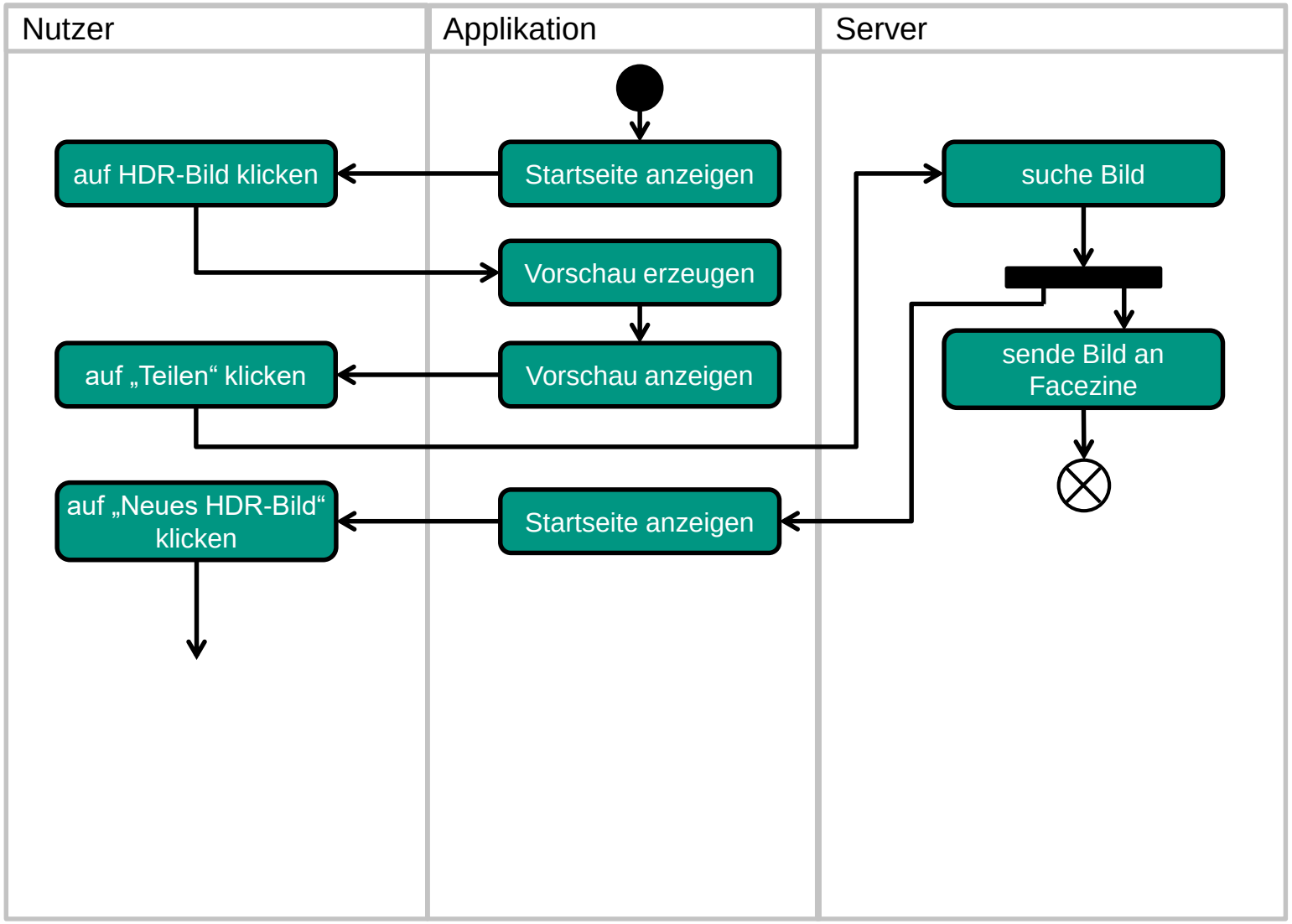
Aufgabe 4 - Aktivitätsdiagramm

Zu Beginn zeigt die Applikation die Startseite an. Der Nutzer kann sich per Klick auf ein HDR-Bild von der Applikation eine Vorschau erzeugen und anzeigen lassen. Daraufhin betätigt der Nutzer die „Teilen“-Schaltfläche, um das Bild auf Facezine hochzuladen. Dies sorgt dafür, dass das entsprechende Bild auf dem Pear-Corp-Zentralserver gesucht und von dort an Facezine gesendet wird. Gleichzeitig mit dem Senden kehrt die Applikation auf die Startseite zurück. Über das Klicken der Schaltfläche „Neues HDR-Bild“ gelangt der Nutzer zu einer Übersicht der auf dem Mobiltelefon gespeicherten Bilder. Der Nutzer wählt drei Bilder einer Bildreihe aus und bestätigt seine Auswahl mit einem Klick auf „HDR-Bild erstellen“. Die Eingabebilder werden anschließend auf den Pear-Corp-Zentralserver hochgeladen und das HDR-Bild erzeugt. Zur Erzeugung des HDR-Bildes rekonstruiert der Server zunächst die Antwortkurve der Kamera der Eingabebilder. Daraufhin kombiniert er die Eingabebilder zu einem HDR-Bild. Hierzu iteriert der Server über die Pixel der Bilder und bestimmt für jeden Pixel den neuen Farbwert. Dazu kombiniert er zunächst die gewichteten Einzelfarbwerte mit der Antwortkurve. Anschließend wird über die Summe der Gewichte normiert. Wurde über alle Pixel iteriert, bildet der Server das Bild in den RGB-Farbraum ab und speichert sowohl das HDR- als auch das RGB-Bild auf dem Server. Die Applikation wartet, solange die Erzeugung nicht beendet ist. Per Klick auf die Schaltfläche „HDR-Bild abholen“ wird dem Nutzer das Vorschaubild und Kaufoptionen angezeigt. Hat der Nutzer eine Kaufoption gewählt, speichert die Applikation das Bild im Bilderordner des Mobiltelefons. Besitzt der Nutzer bereits ein Monatsabonnement, sollen keine Kaufoptionen, sondern direkt eine Schaltfläche „HDR-Bild jetzt speichern“ angezeigt werden.

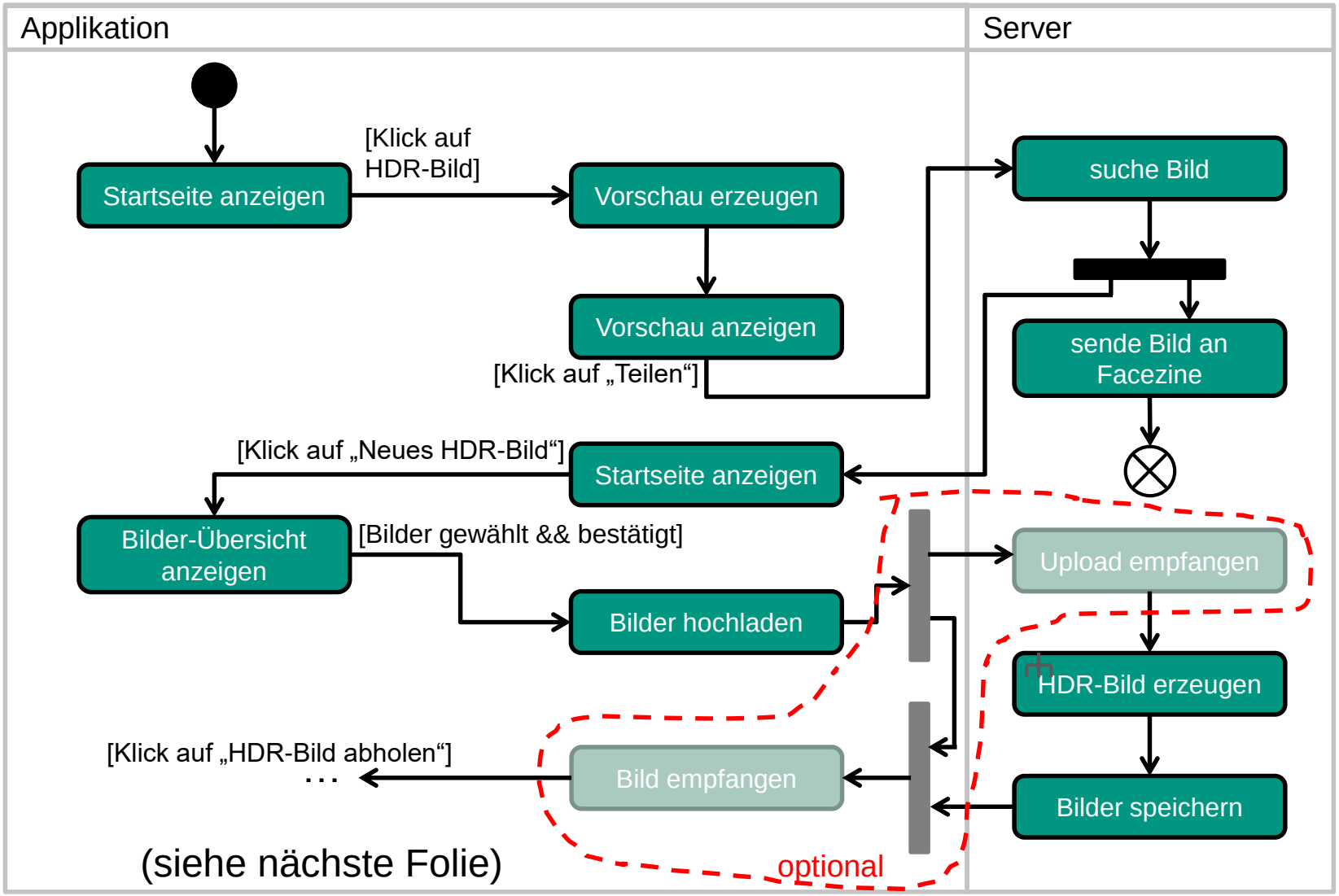
Aufgabe 4 - Aktivitätsdiagramm

Zu Beginn zeigt die Applikation die Startseite an. Der Nutzer kann sich per Klick auf ein HDR-Bild von der Applikation eine Vorschau erzeugen und anzeigen lassen. Daraufhin betätigt der Nutzer die „Teilen“-Schaltfläche, um das Bild auf Facezine hochzuladen. Dies sorgt dafür, dass das entsprechende Bild auf dem Pear-Corp-Zentralserver gesucht und von dort an Facezine gesendet wird. Gleichzeitig mit dem Senden kehrt die Applikation auf die Startseite zurück. Über das Klicken der Schaltfläche „Neues HDR-Bild“ gelangt der Nutzer zu einer Übersicht der auf dem Mobiltelefon gespeicherten Bilder. Der Nutzer wählt drei Bilder einer Bildreihe aus und bestätigt seine Auswahl mit einem Klick auf „HDR-Bild erstellen“. Die Eingabebilder werden anschließend auf den Pear-Corp-Zentralserver hochgeladen und das HDR-Bild erzeugt. Zur Erzeugung des HDR-Bildes rekonstruiert der Server zunächst die Antwortkurve der Kamera der Eingabebilder. Daraufhin kombiniert er die Eingabebilder zu einem HDR-Bild. Hierzu iteriert der Server über die Pixel der Bilder und bestimmt für jeden Pixel den neuen Farbwert. Dazu kombiniert er zunächst die gewichteten Einzelfarbwerte mit der Antwortkurve. Anschließend wird über die Summe der Gewichte normiert. Wurde über alle Pixel iteriert, bildet der Server das Bild in den RGB-Farbraum ab und speichert sowohl das HDR- als auch das RGB-Bild auf dem Server. Die Applikation wartet, solange die Erzeugung nicht beendet ist. Per Klick auf die Schaltfläche „HDR-Bild abholen“ wird dem Nutzer das Vorschaubild und Kaufoptionen angezeigt. Hat der Nutzer eine Kaufoption gewählt, speichert die Applikation das Bild im Bilderordner des Mobiltelefons. Besitzt der Nutzer bereits ein Monatsabonnement, sollen keine Kaufoptionen, sondern direkt eine Schaltfläche „HDR-Bild jetzt speichern“ angezeigt werden.

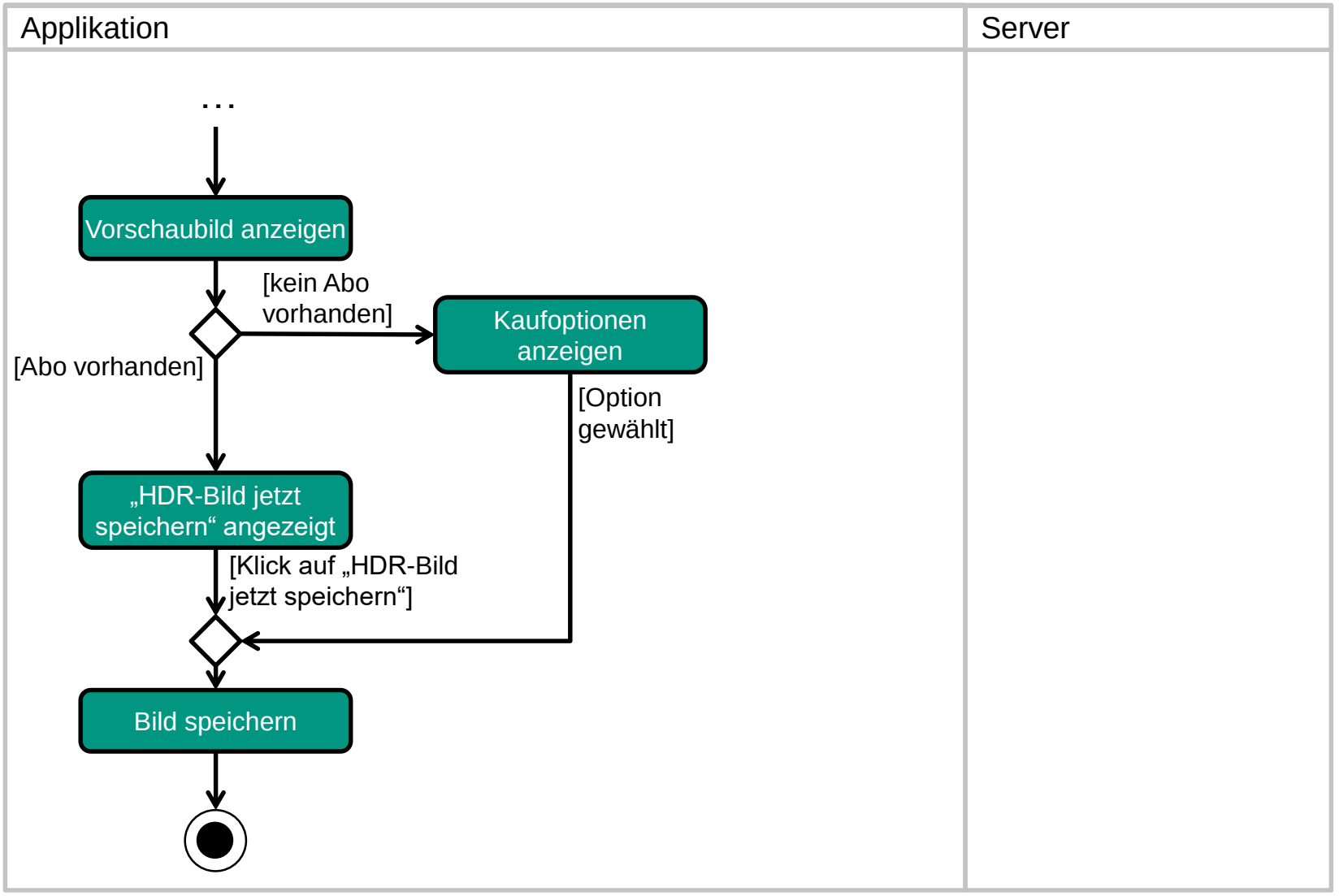
Aufgabe 4 – Aktivitätsdiagramm Teil 1



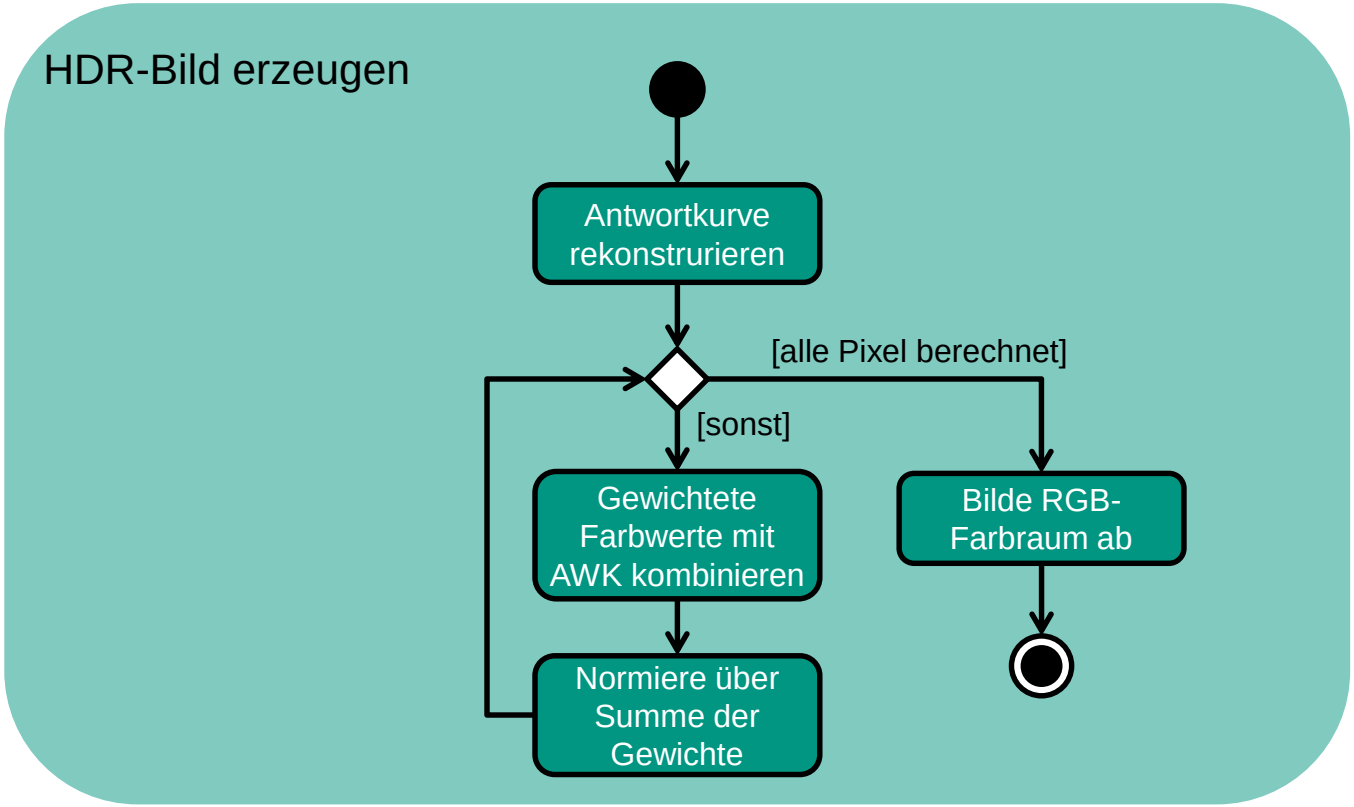
Aufgabe 4 – Aktivitätsdiagramm Teil 1



Aufgabe 4 – Aktivitätsdiagramm Teil 2



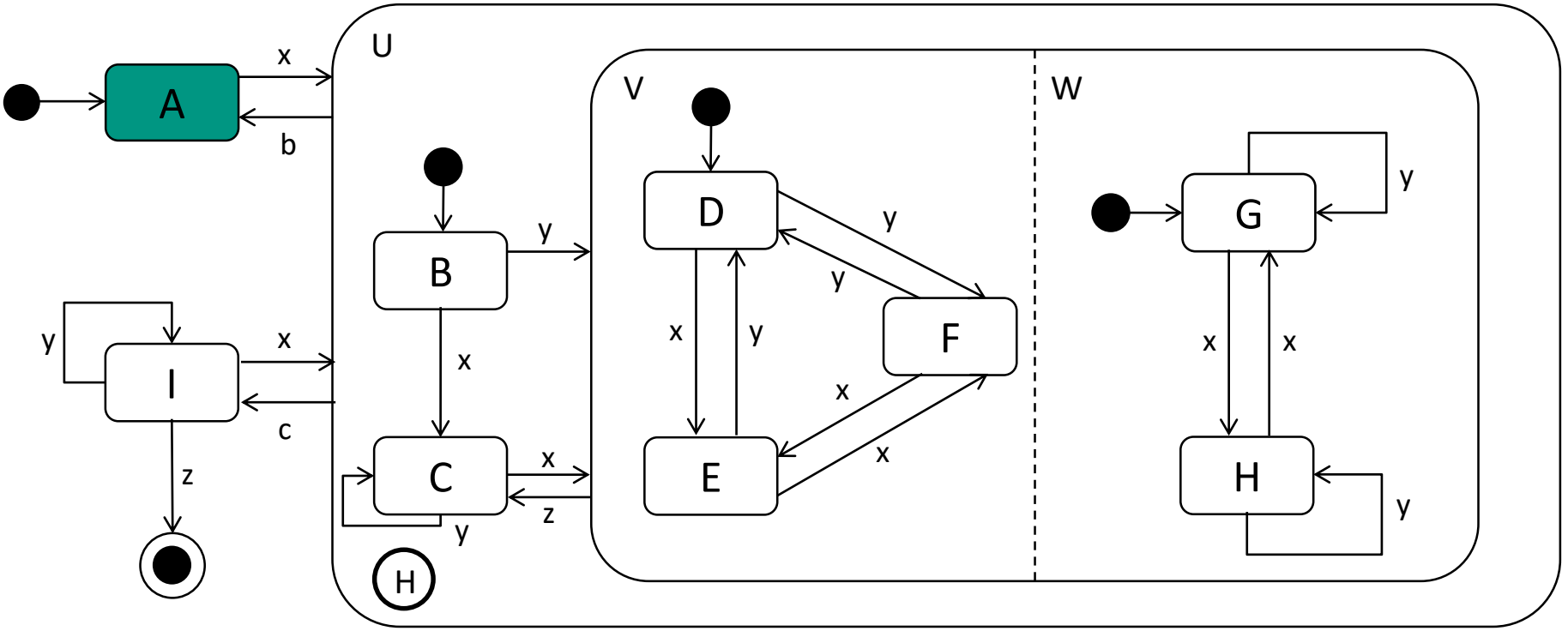
Aufgabe 4 – Aktivitätsdiagramm Teil 3



UML-Zustandsdiagramm

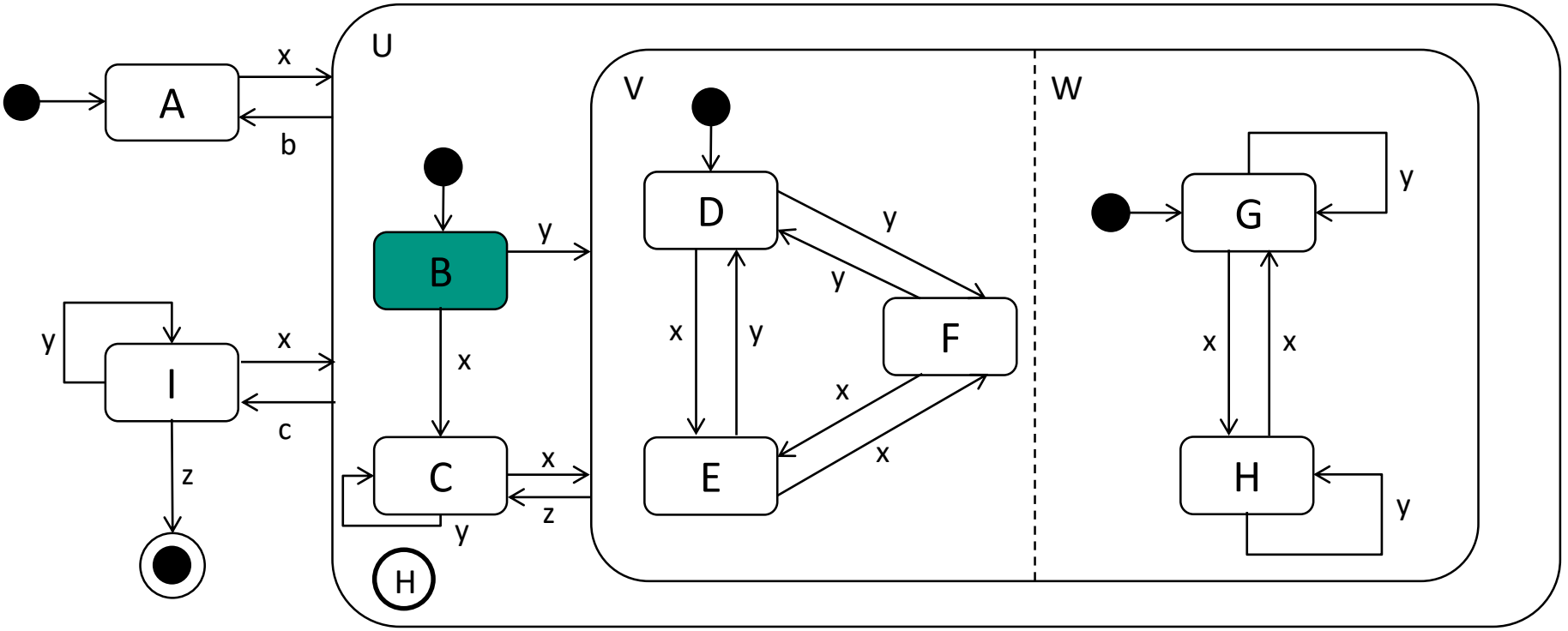
AUFGABE 5

Aufgabe 5 - Zustandsdiagramm



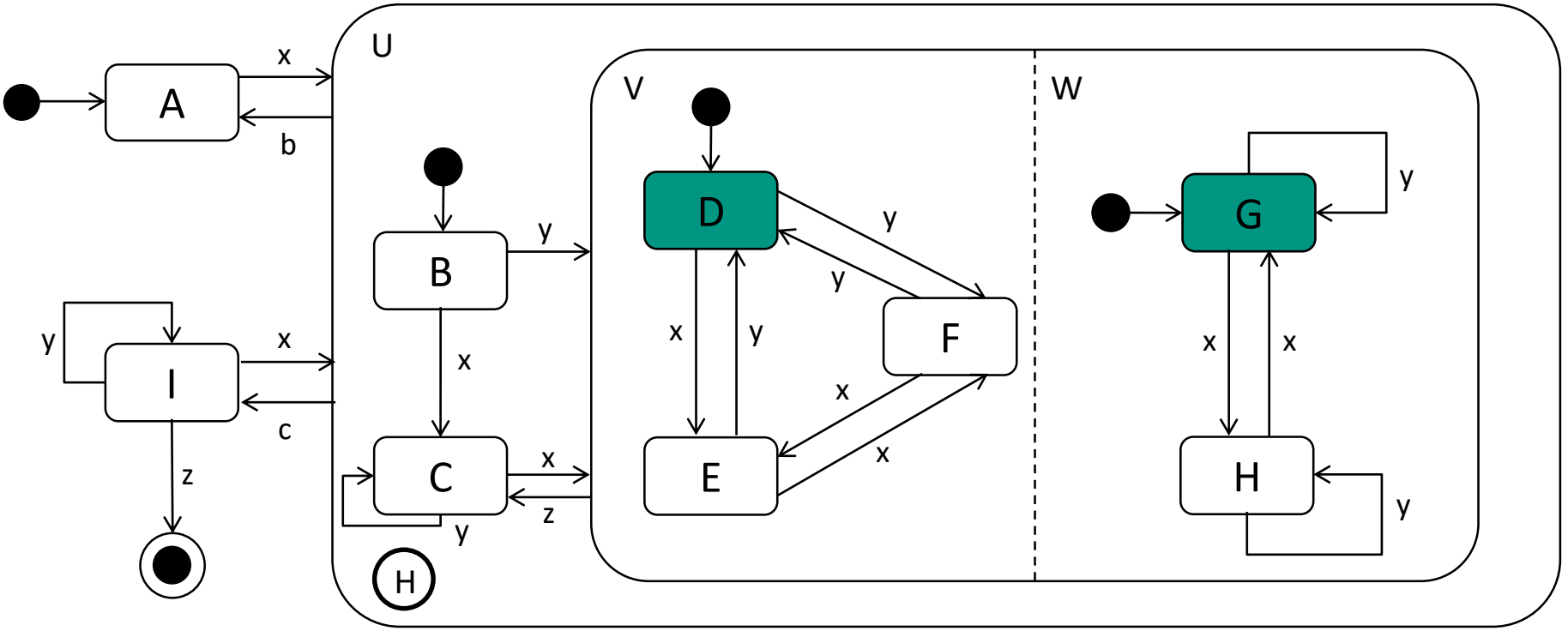
a) x, y, x, y, y, z, b, x, y, c, y, x

Aufgabe 5 - Zustandsdiagramm



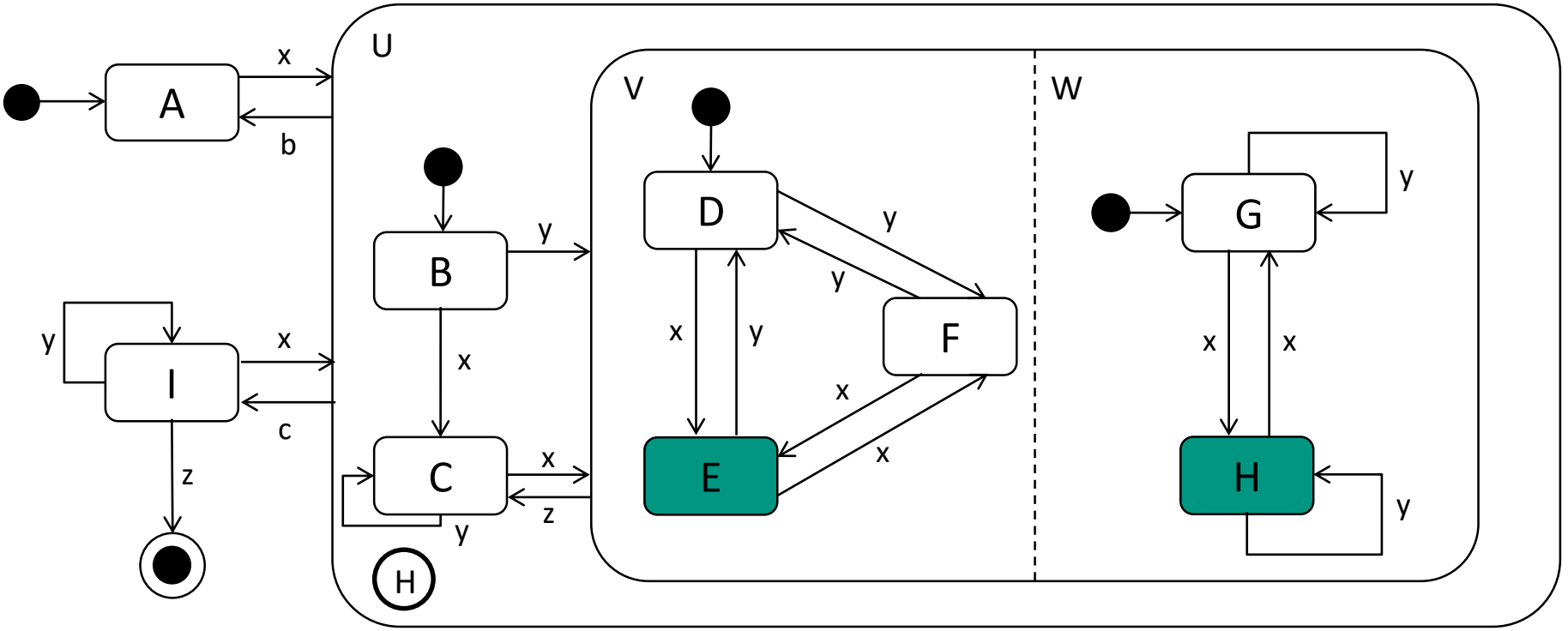
a) x, y, x, y, y, z, b, x, y, c, y, x

Aufgabe 5 - Zustandsdiagramm



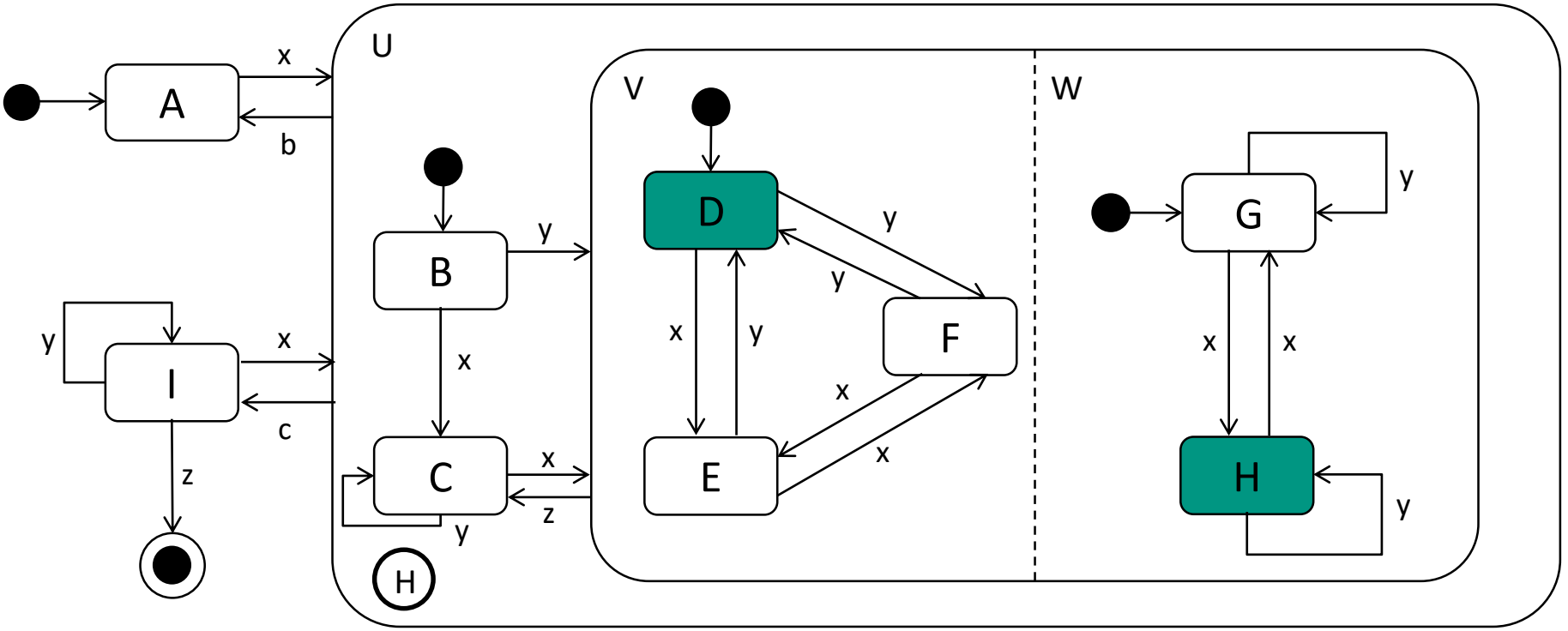
a) x, y, x, y, y, z, b, x, y, c, y, x

Aufgabe 5 - Zustandsdiagramm



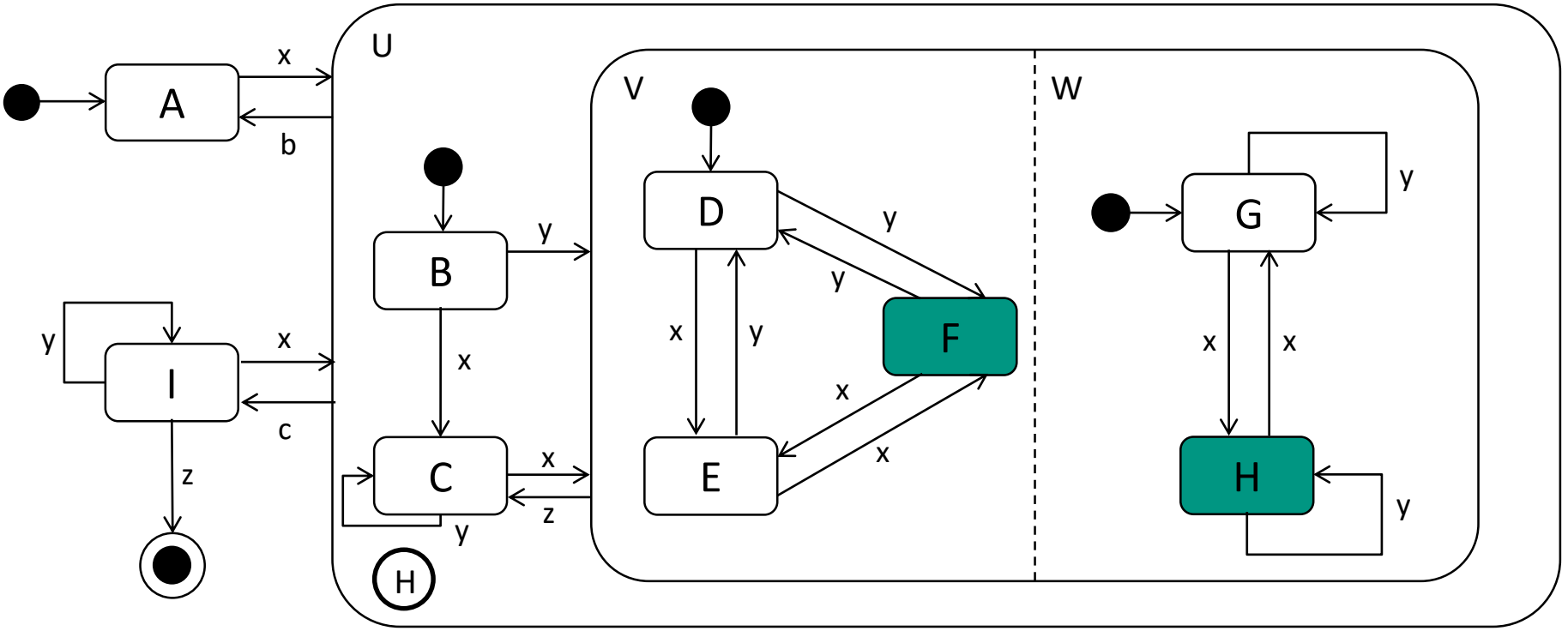
a) x, y, x, y, y, z, b, x, y, c, y, x

Aufgabe 5 - Zustandsdiagramm



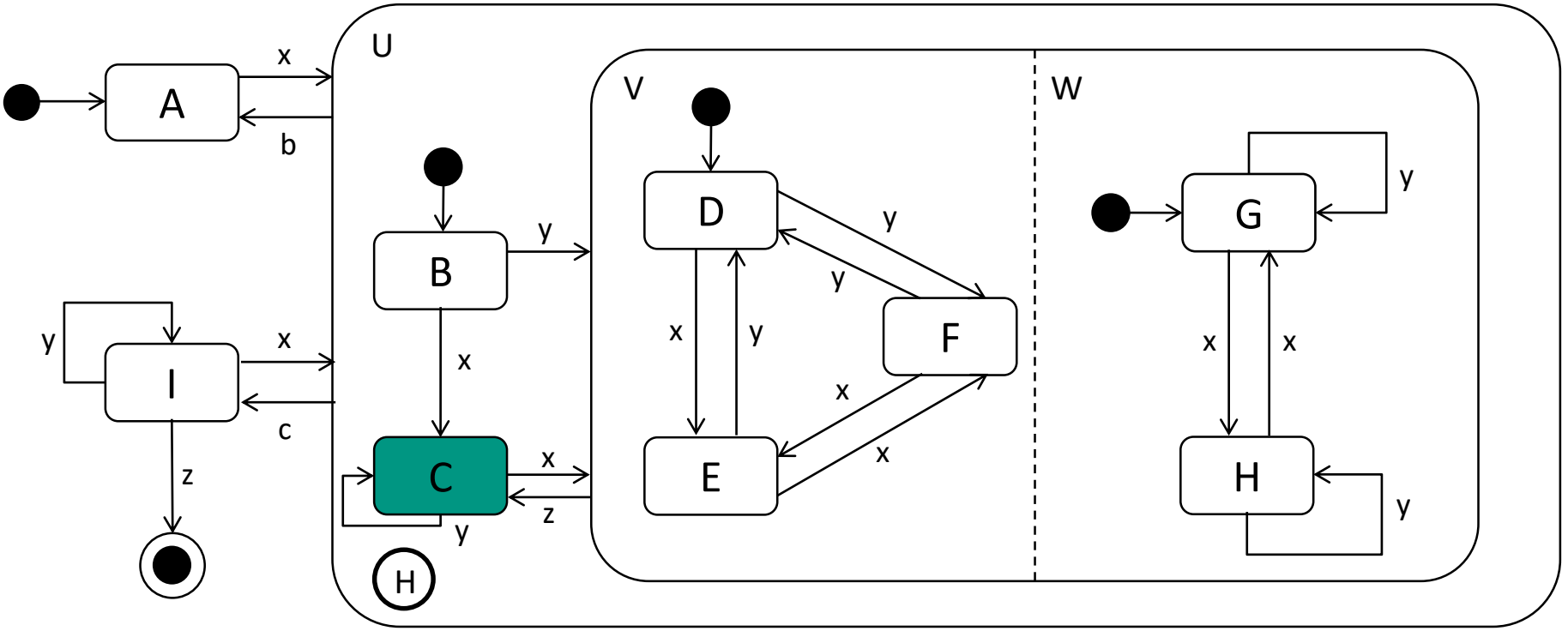
a) x, y, x, y, y, z, b, x, y, c, y, x

Aufgabe 5 - Zustandsdiagramm



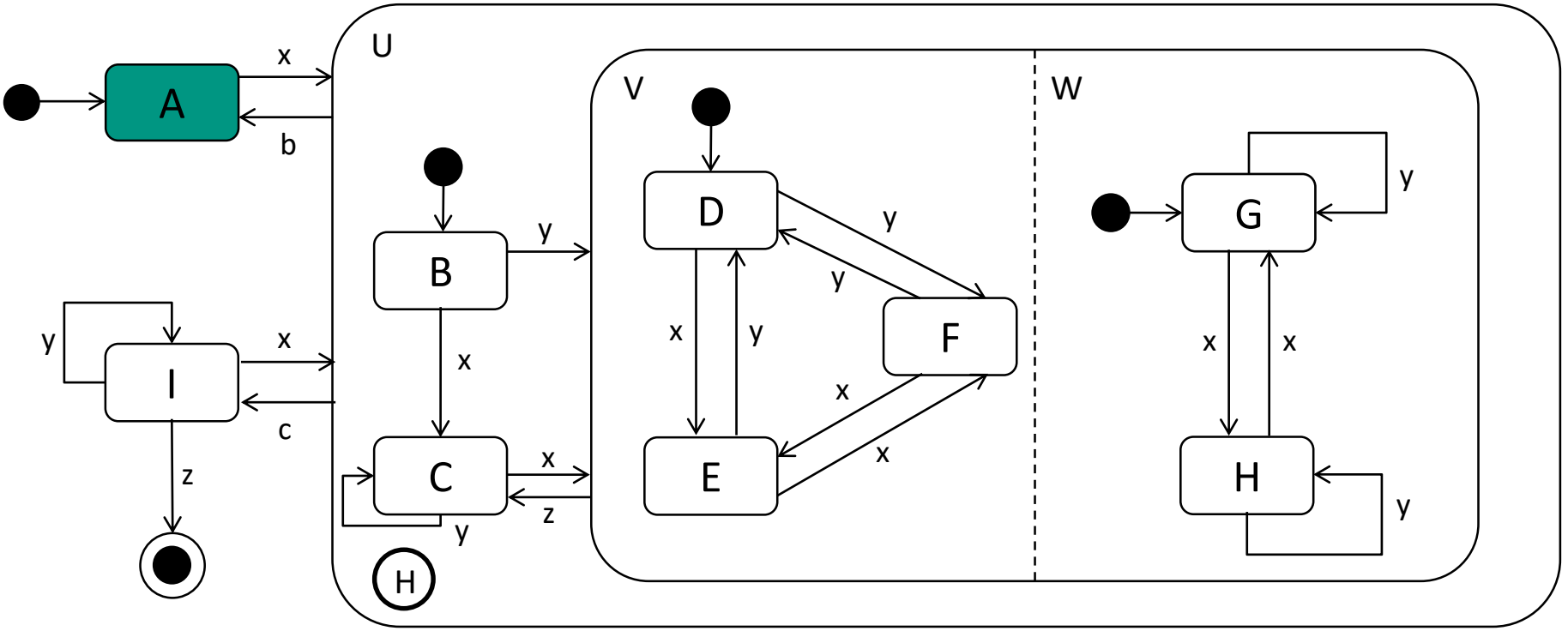
a) x, y, x, y, y, z, b, x, y, c, y, x

Aufgabe 5 - Zustandsdiagramm



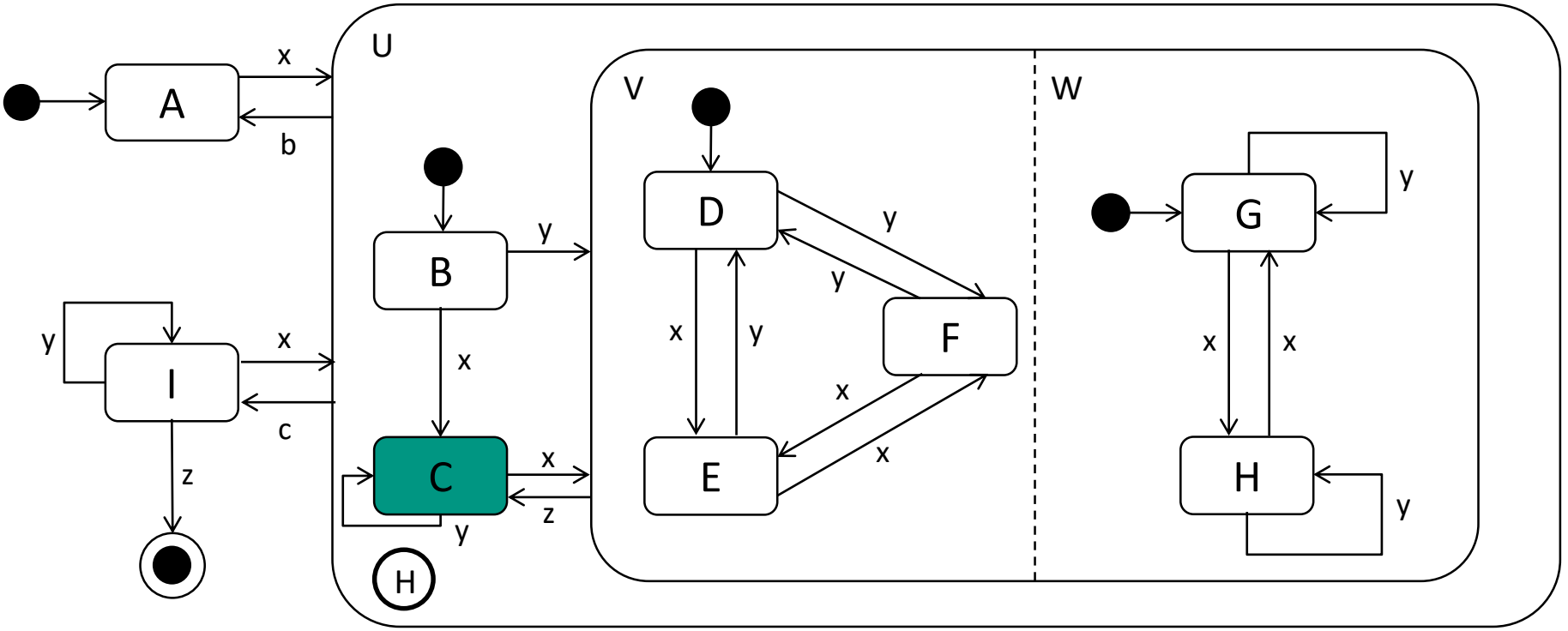
a) x, y, x, y, y, z, b, x, y, c, y, x

Aufgabe 5 - Zustandsdiagramm



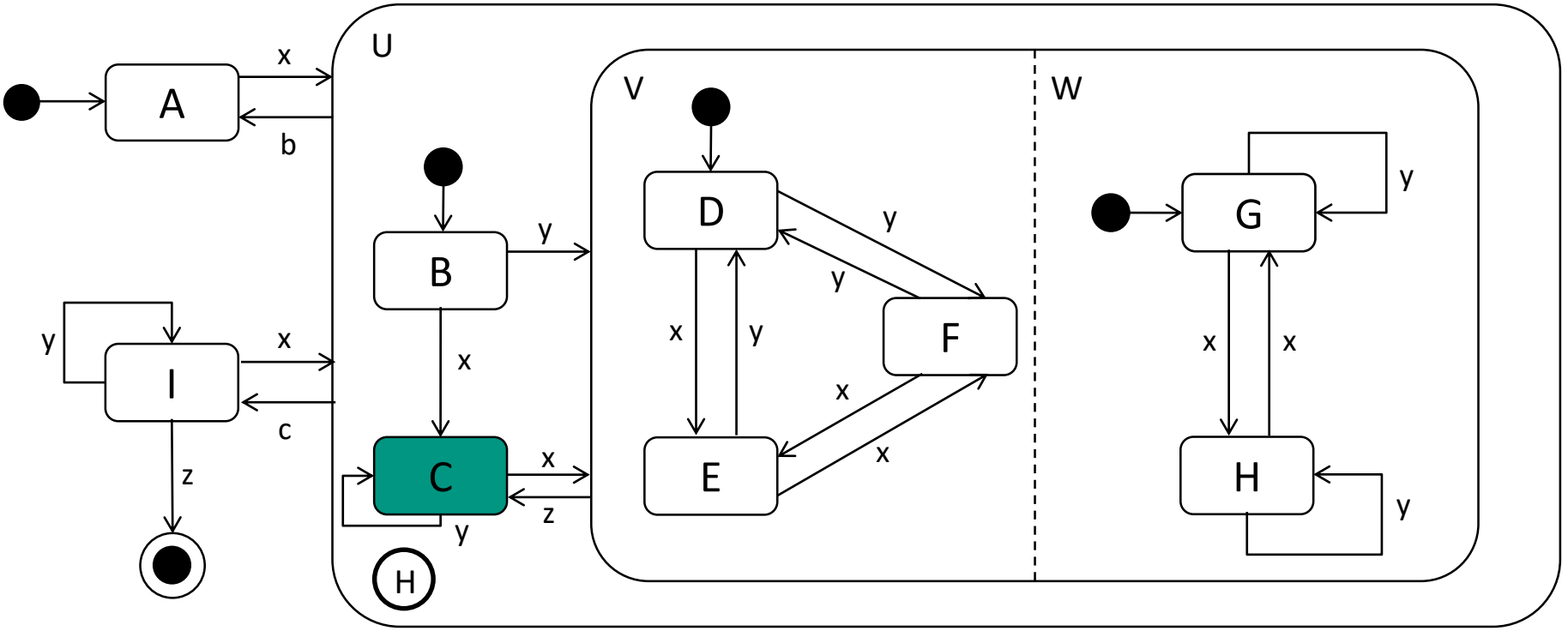
a) x, y, x, y, y, z, b, x, y, c, y, x

Aufgabe 5 - Zustandsdiagramm



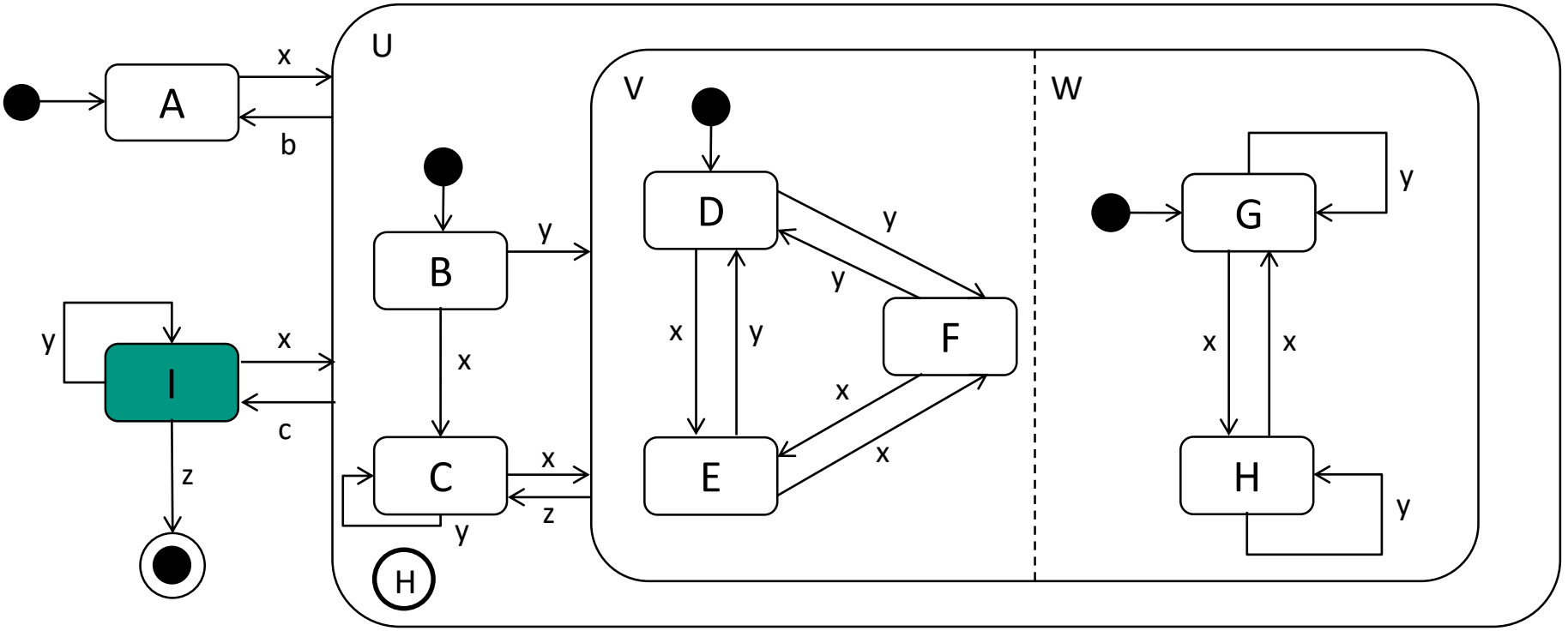
a) x, y, x, y, y, z, b, x, y, c, y, x

Aufgabe 5 - Zustandsdiagramm



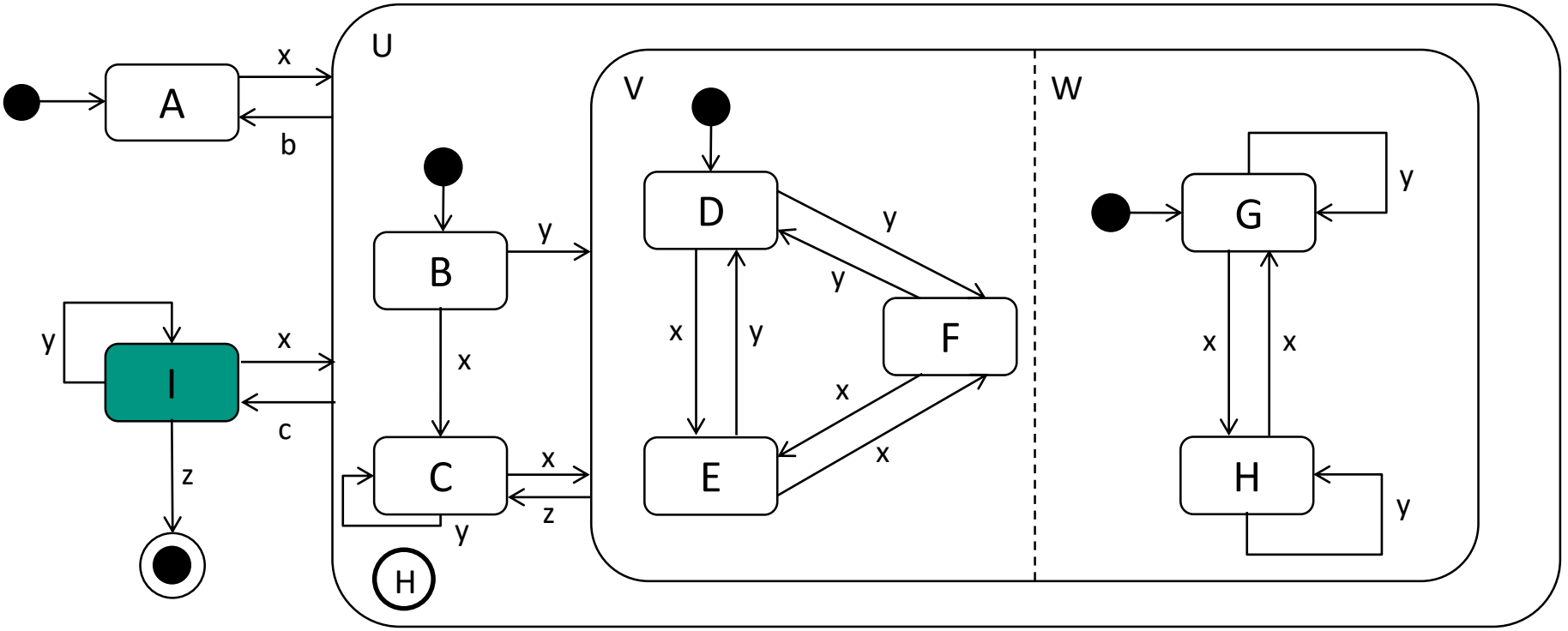
a) x, y, x, y, y, z, b, x, y, c, y, x

Aufgabe 5 - Zustandsdiagramm



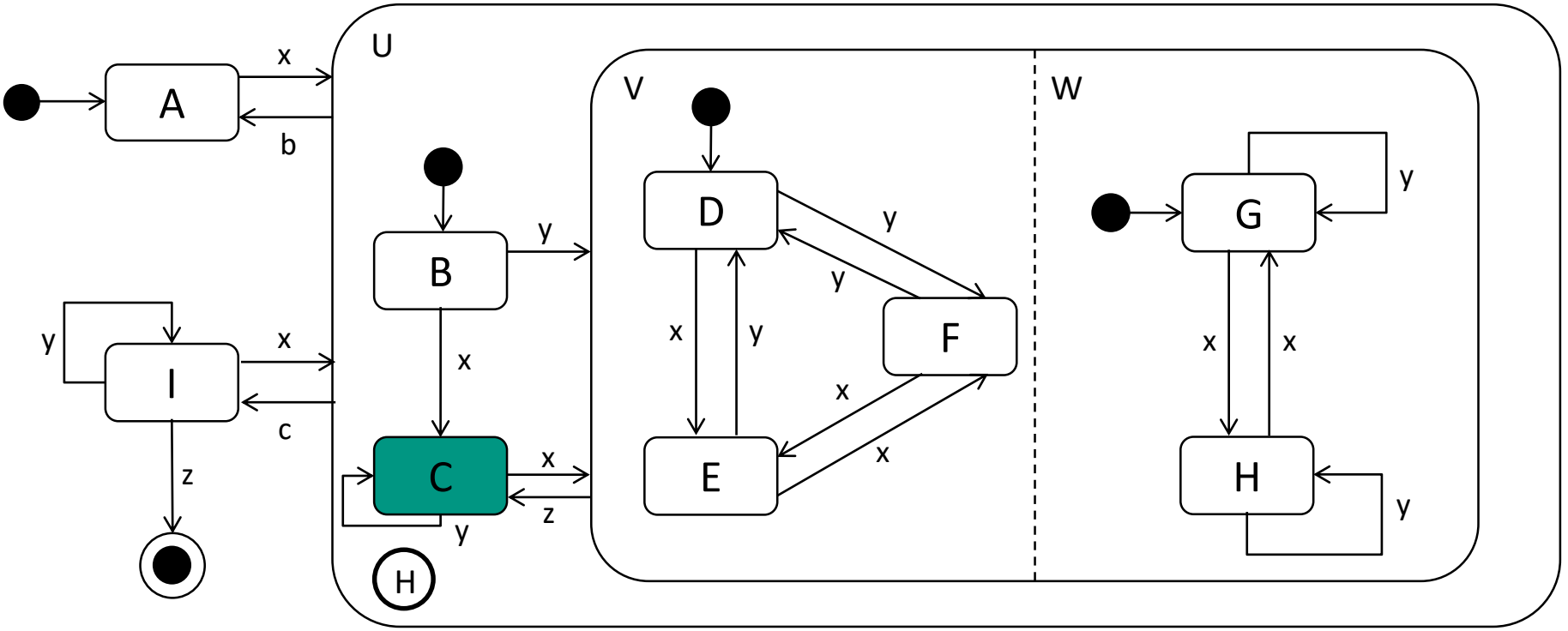
a) x, y, x, y, y, z, b, x, y, c, y, x

Aufgabe 5 - Zustandsdiagramm



a) x, y, x, y, y, z, b, x, y, c, y, x

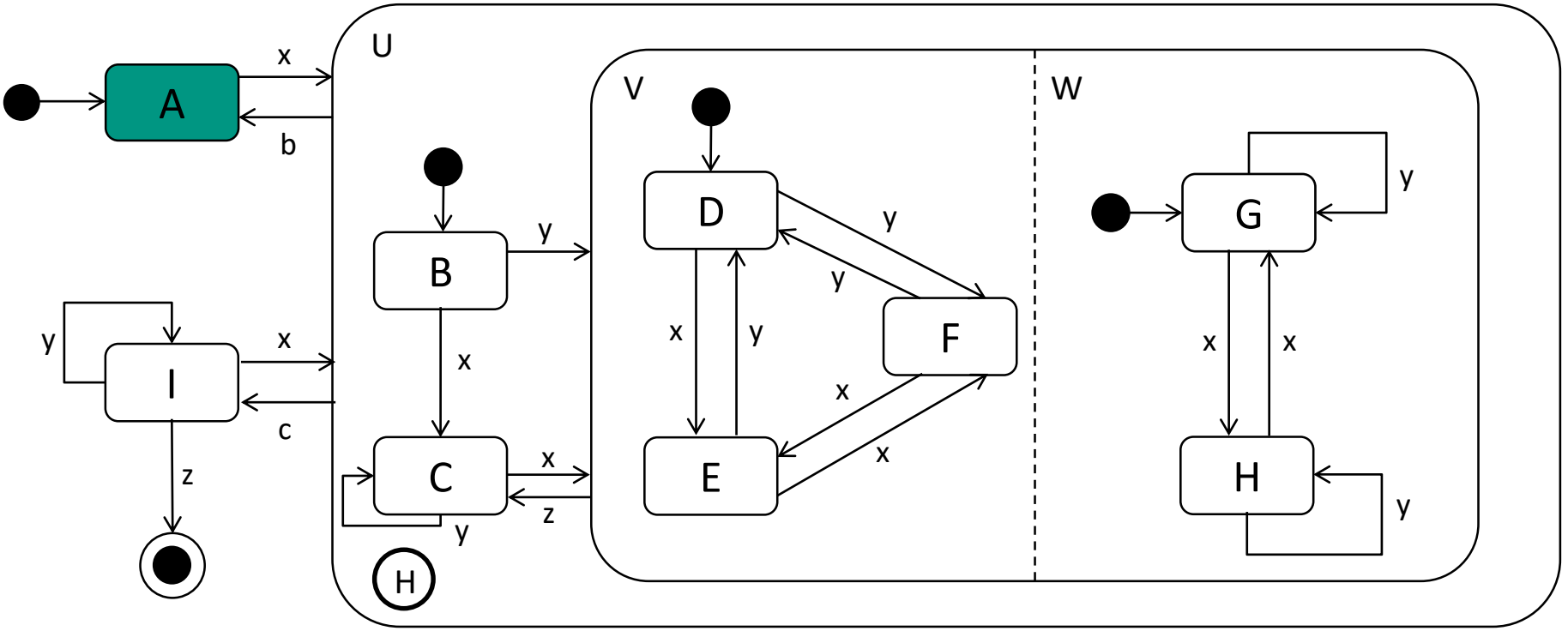
Aufgabe 5 - Zustandsdiagramm



a) x, y, x, y, y, z, b, x, y, c, y, x

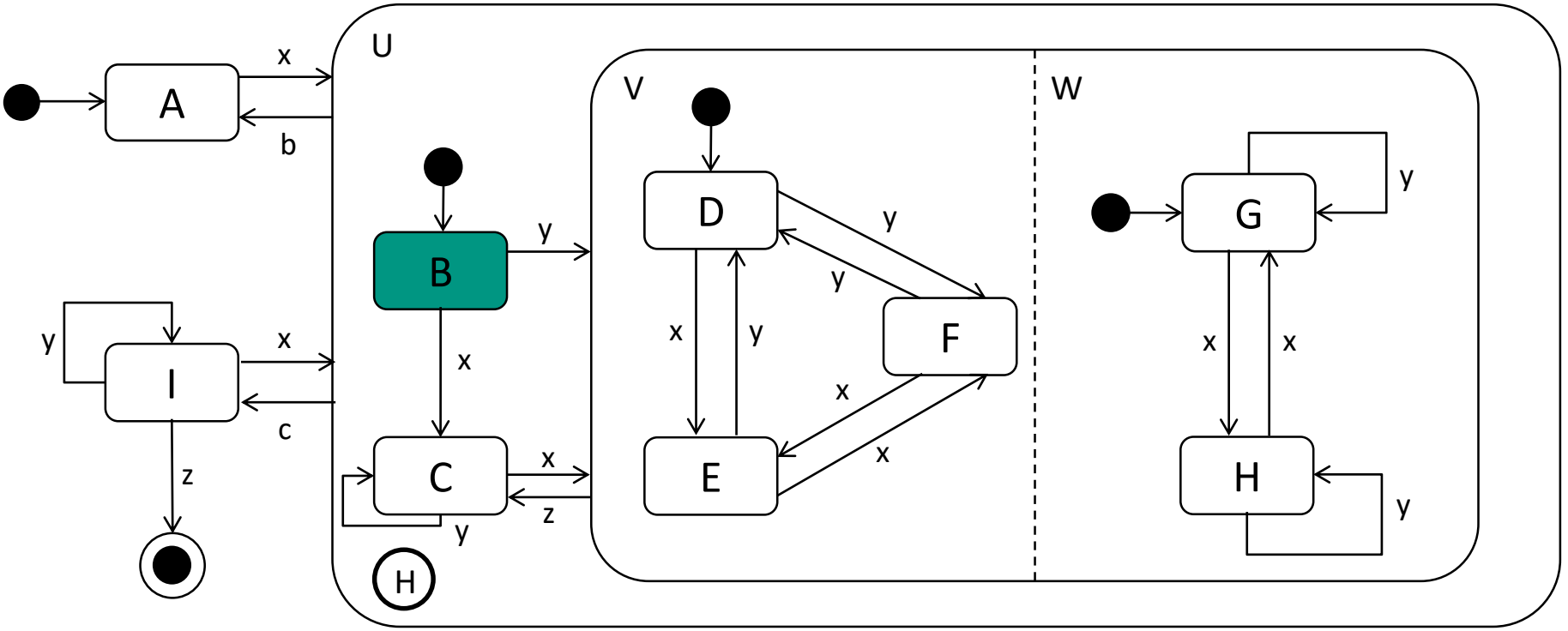
A→B→DxG→ExH→DxH→FxH→C→A→C→C→I→I→C

Aufgabe 5 - Zustandsdiagramm



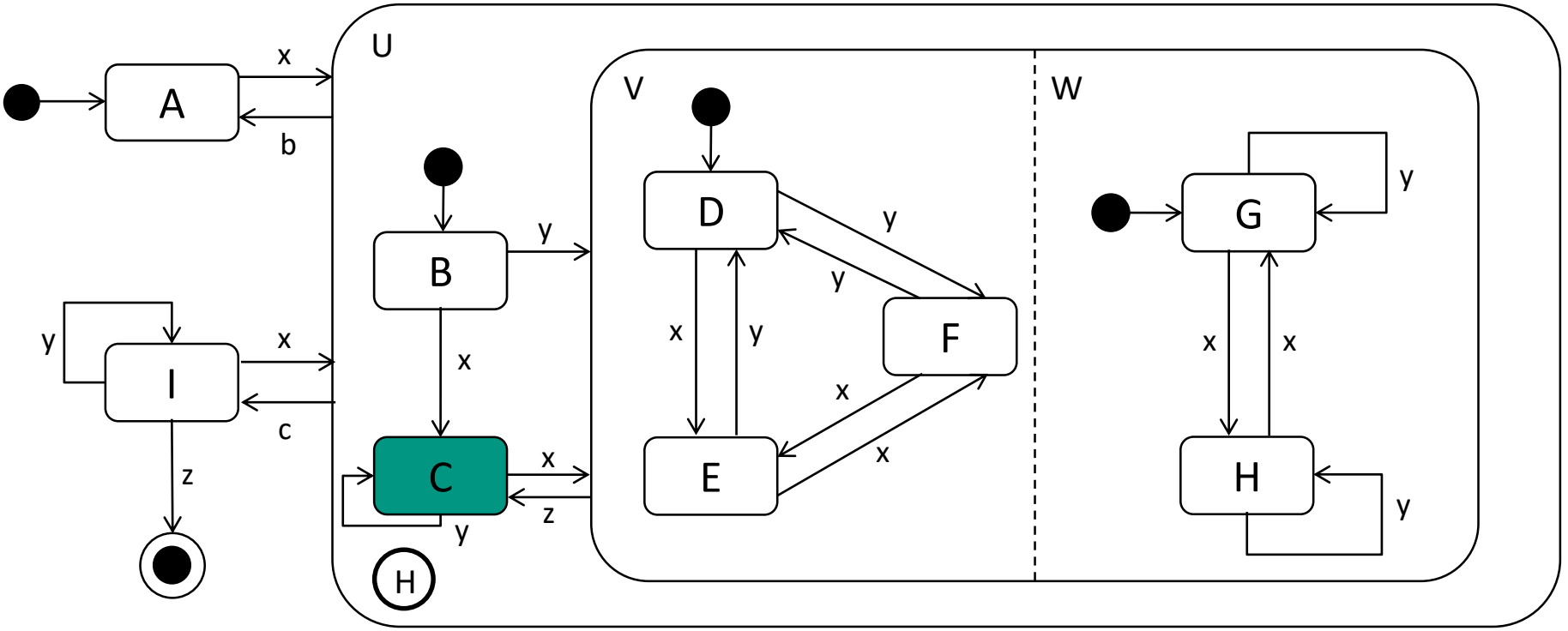
b) x, x, x, x, x, z, x, y, x, x

Aufgabe 5 - Zustandsdiagramm



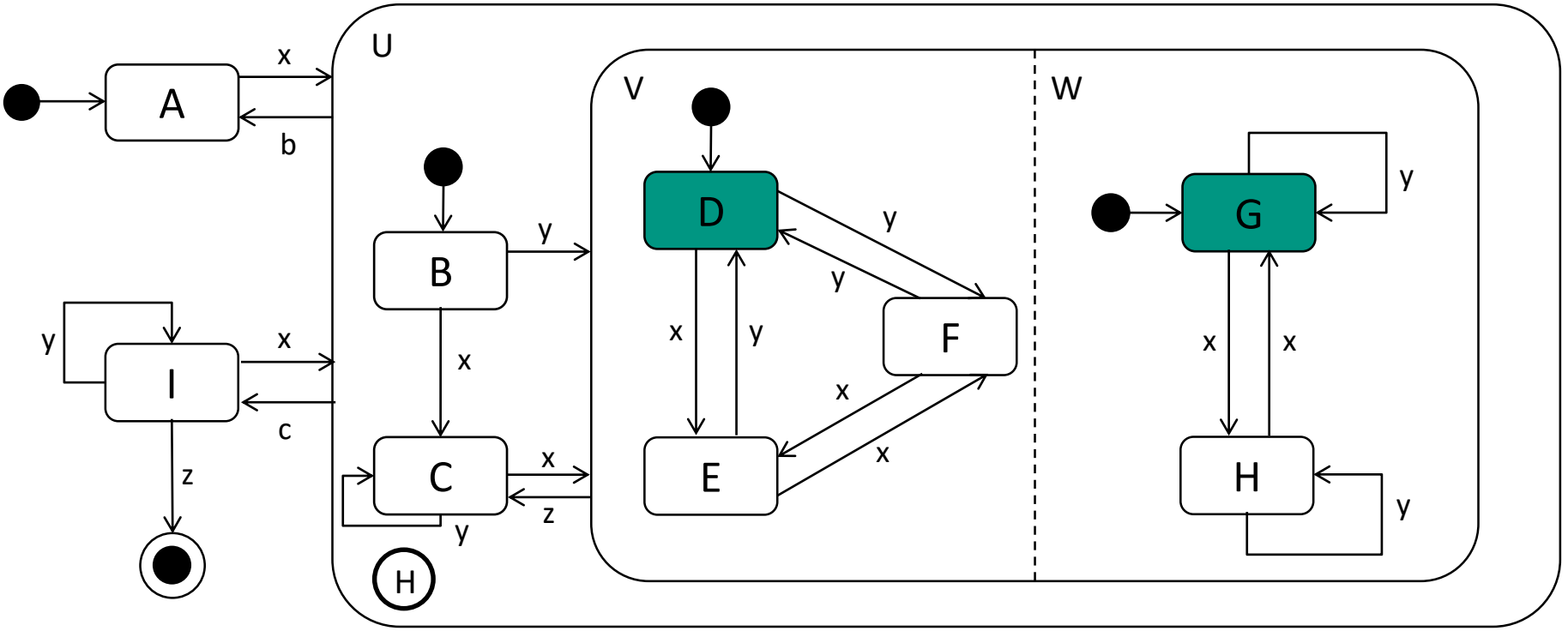
b) x, x, x, x, x, z, x, y, x, x

Aufgabe 5 - Zustandsdiagramm



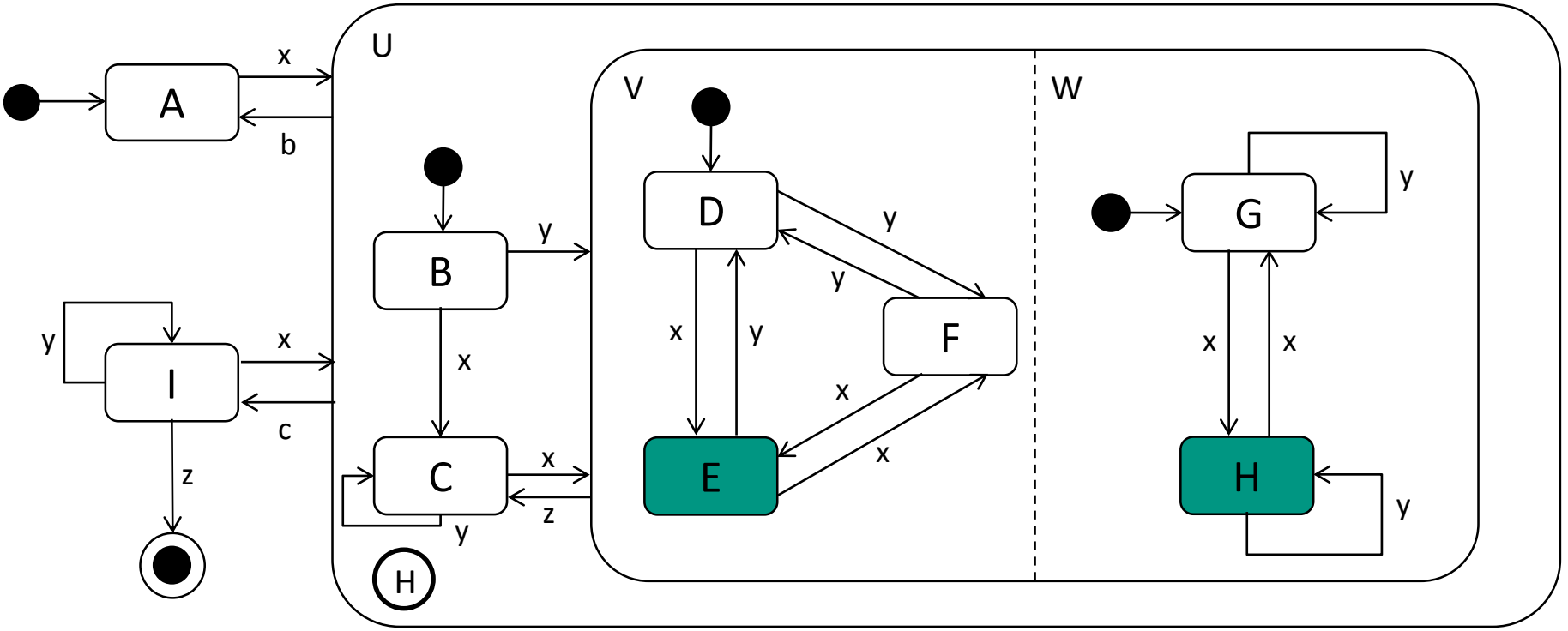
b) x, x, x, x, x, z, x, y, x, x

Aufgabe 5 - Zustandsdiagramm



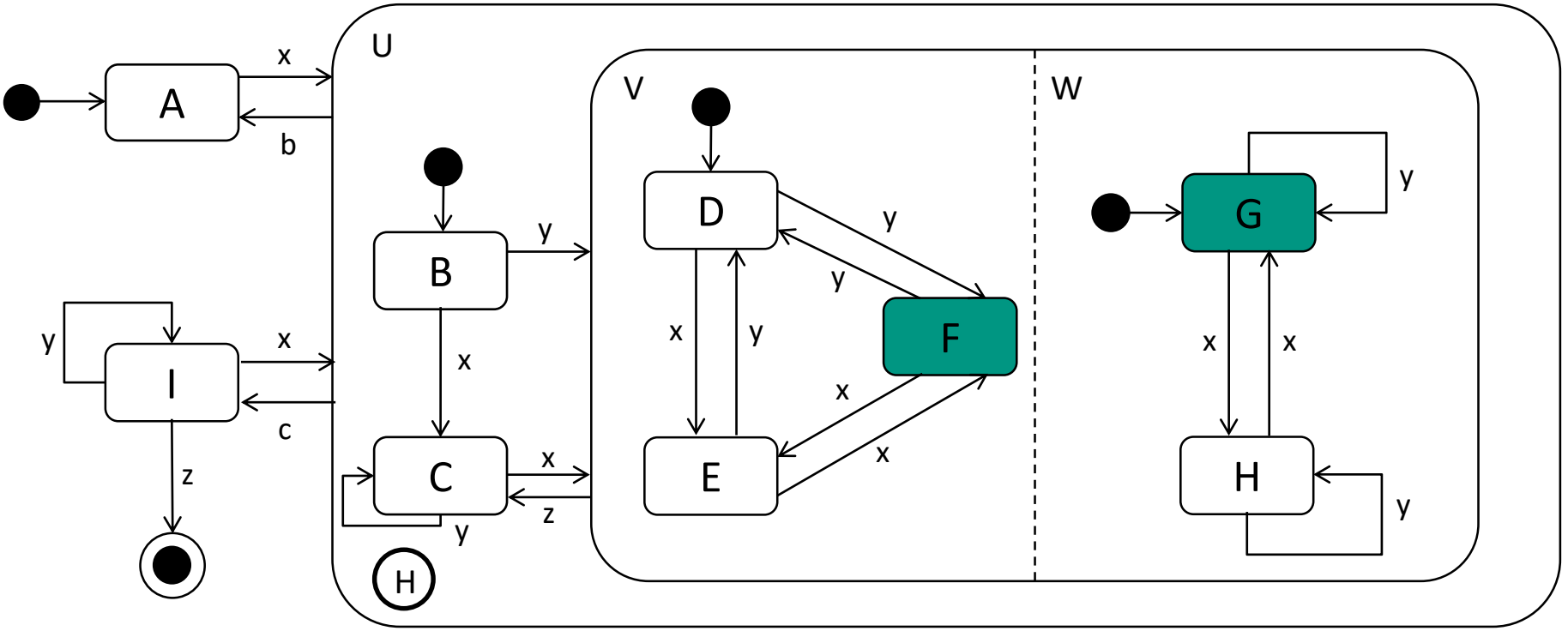
b) x, x, x, x, x, z, x, y, x, x

Aufgabe 5 - Zustandsdiagramm



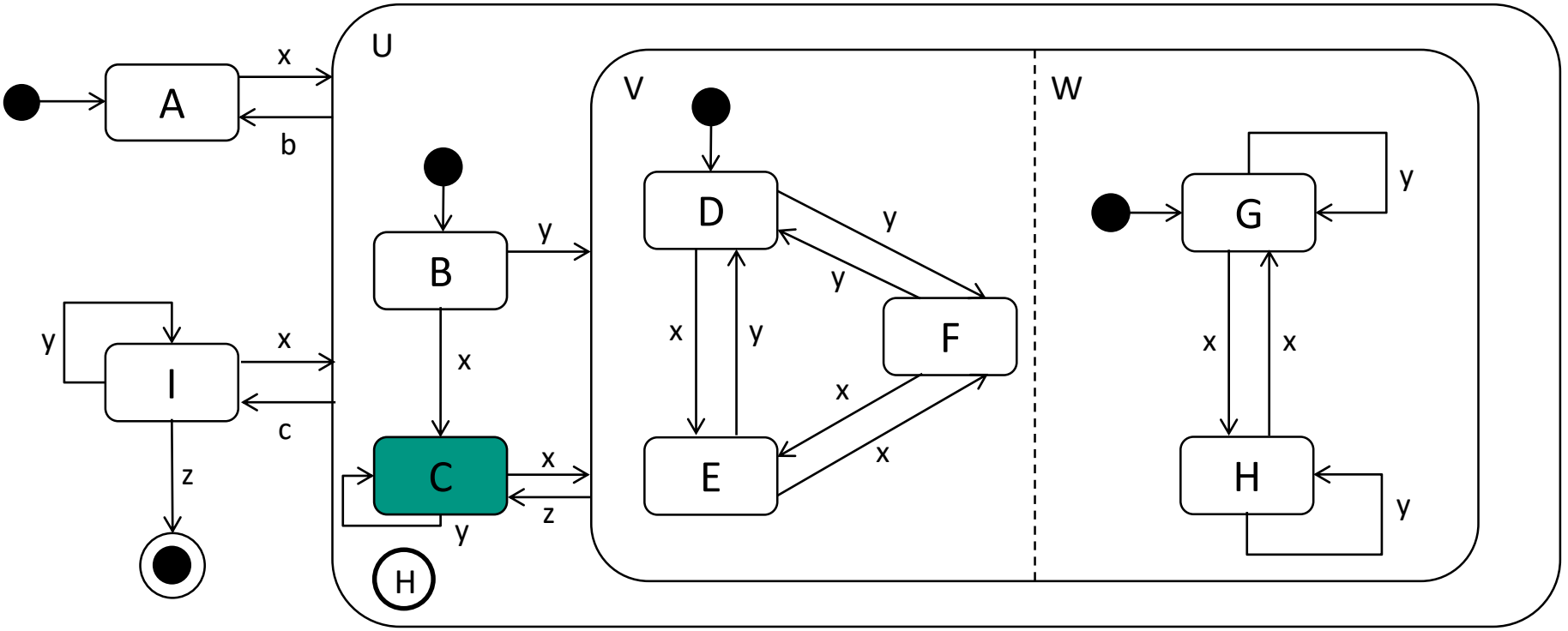
b) x, x, x, x, x, z, x, y, x, x

Aufgabe 5 - Zustandsdiagramm



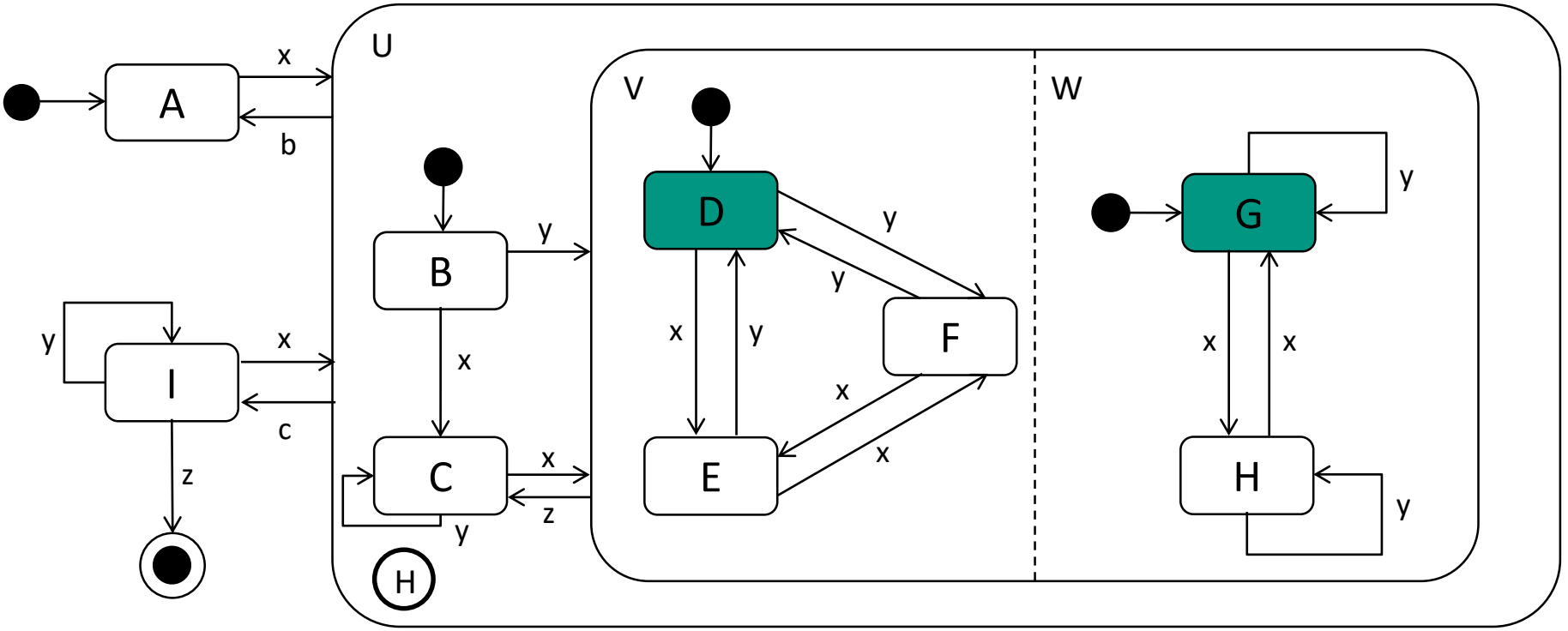
b) x, x, x, x, x, z, x, y, x, x

Aufgabe 5 - Zustandsdiagramm



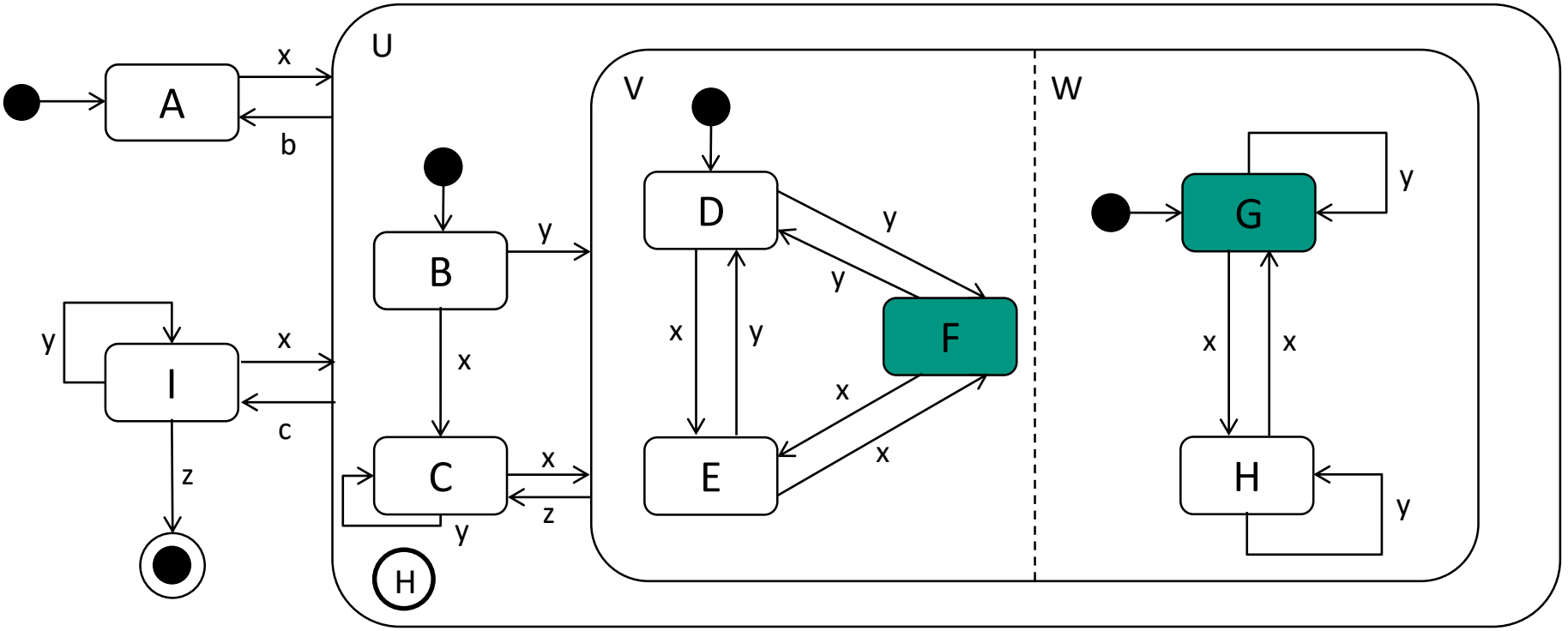
b) x, x, x, x, x, z, x, y, x, x

Aufgabe 5 - Zustandsdiagramm



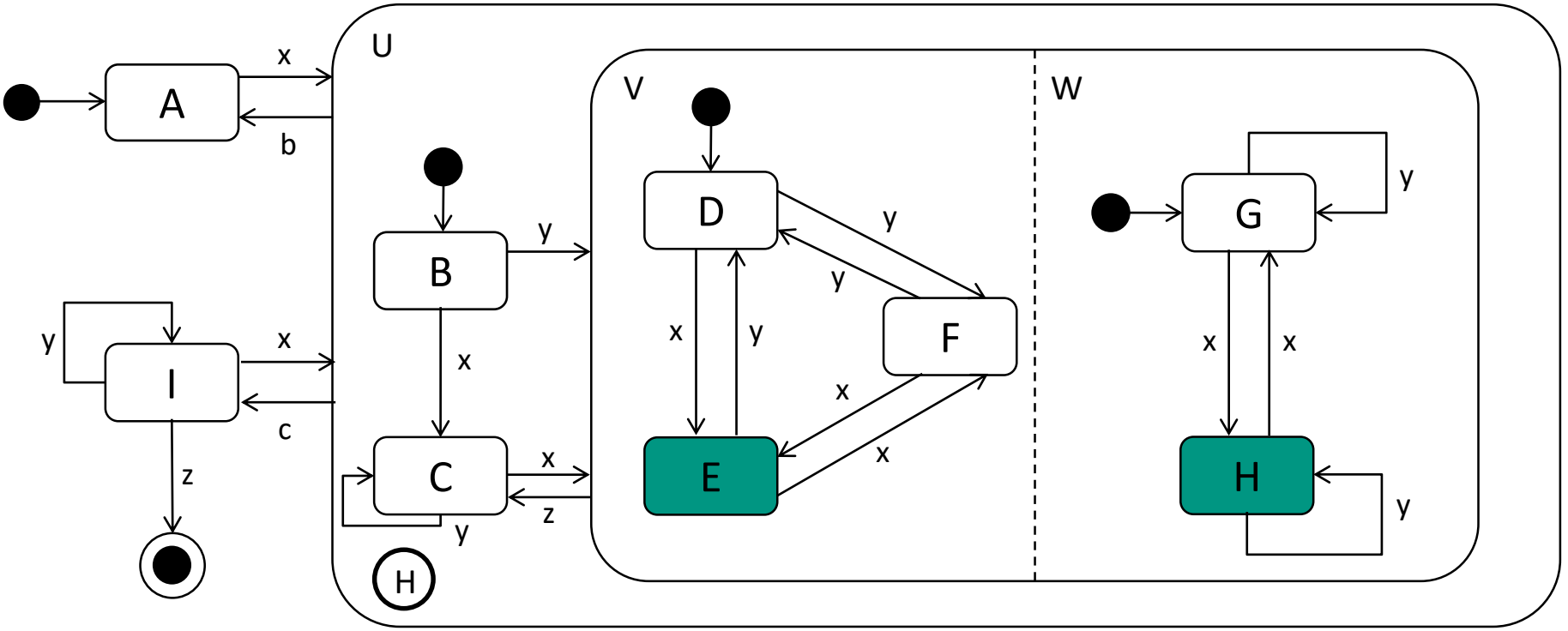
b) x, x, x, x, x, z, x, y, x, x

Aufgabe 5 - Zustandsdiagramm



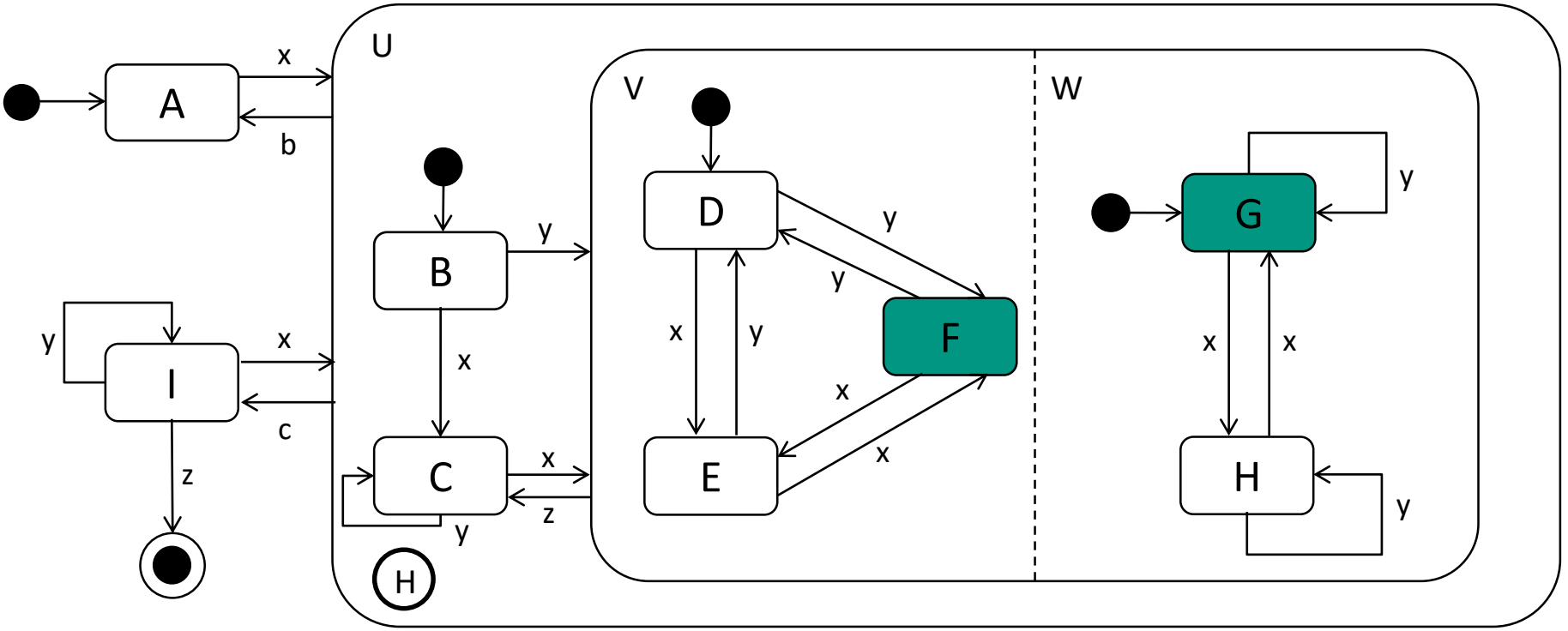
b) x, x, x, x, x, z, x, y, x, x

Aufgabe 5 - Zustandsdiagramm



b) x, x, x, x, x, z, x, y, x, x

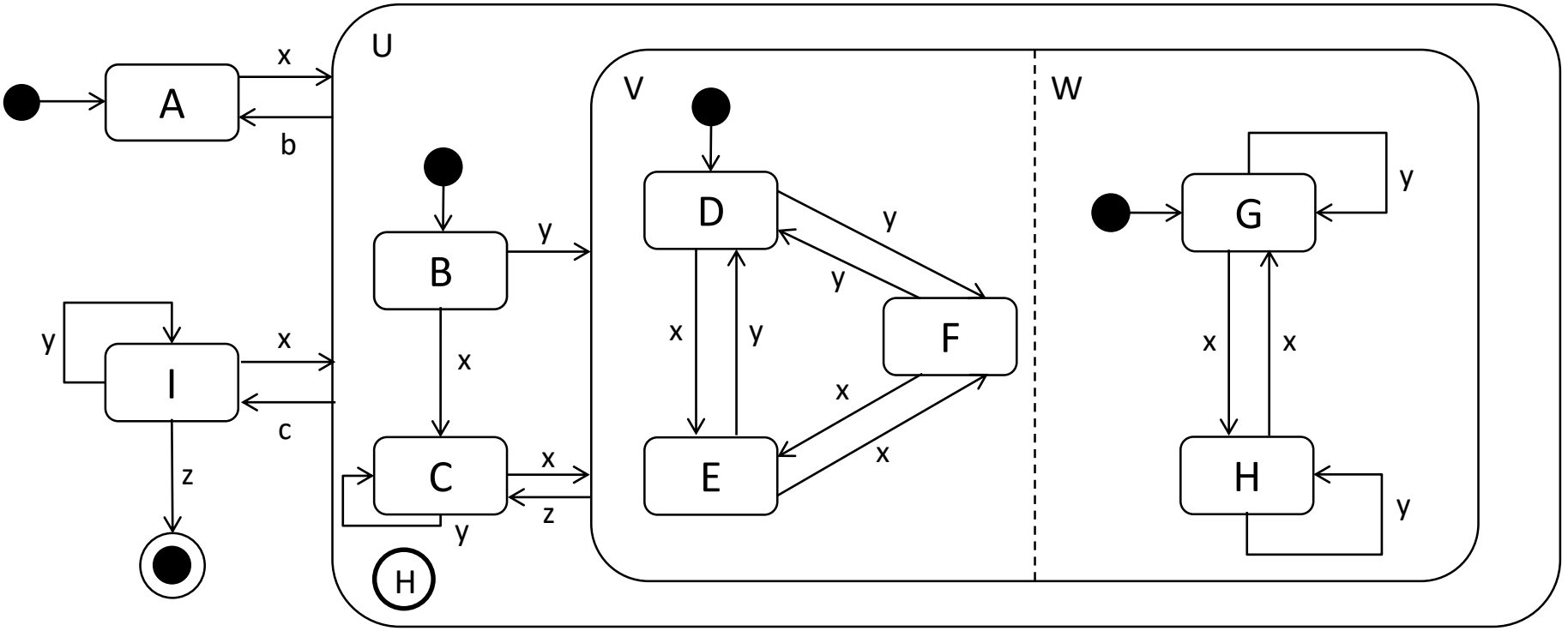
Aufgabe 5 - Zustandsdiagramm



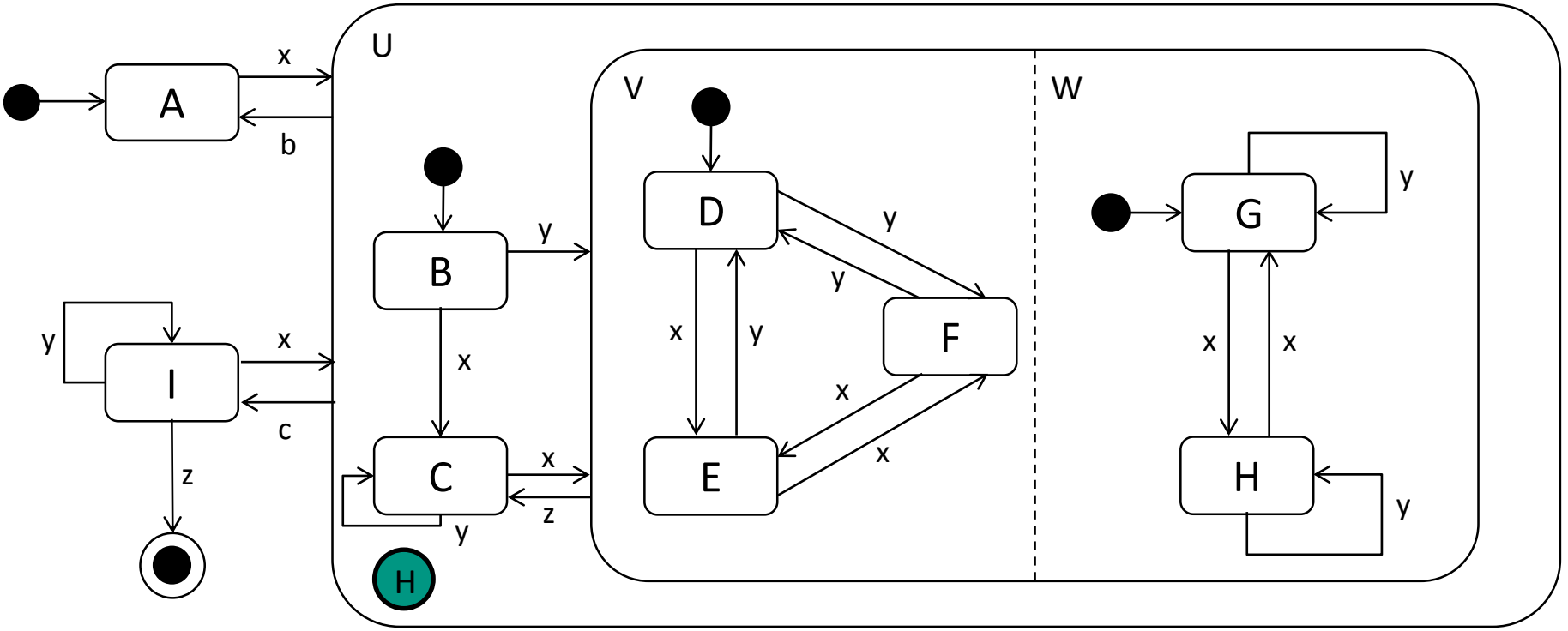
b) x, x, x, x, x, z, x, y, x, x

A→B→C→DxG→ExH→FxG→C→DxG→FxG→ExH→FxG

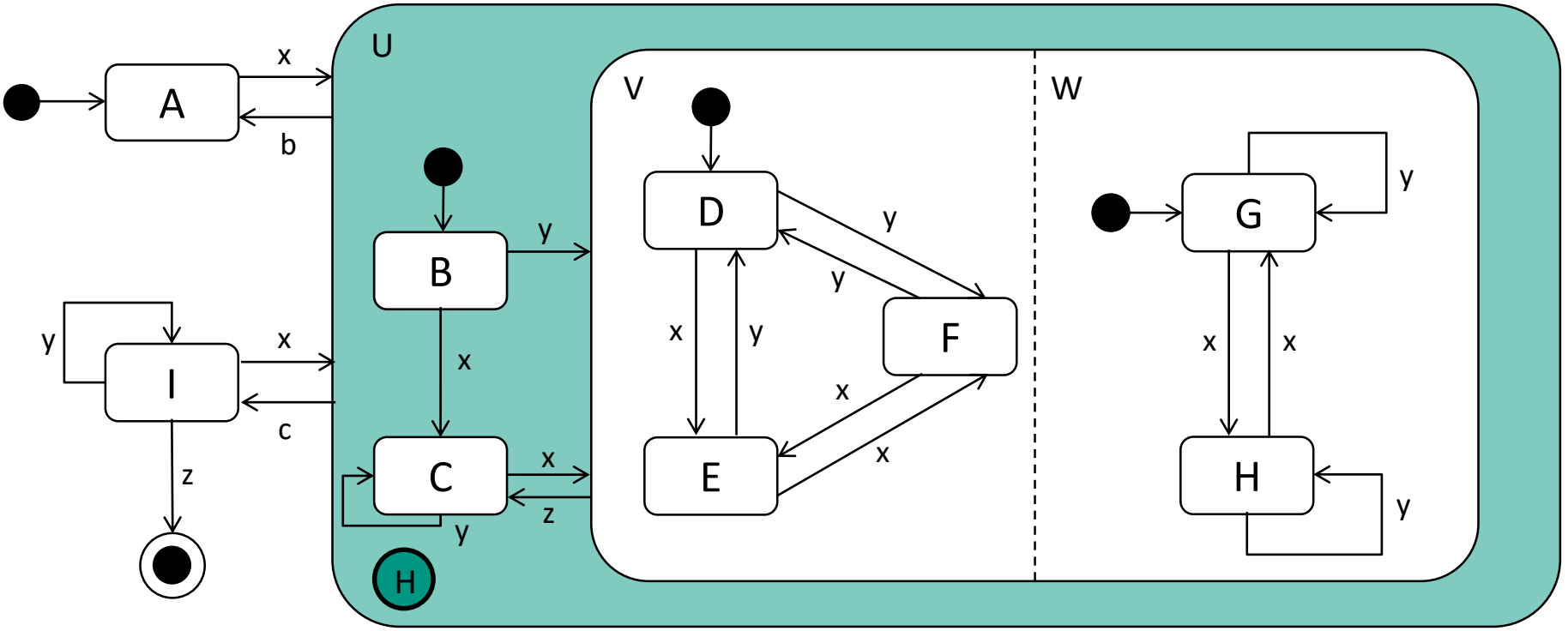
Aufgabe 5 – Zustandsdiagramm umwandeln



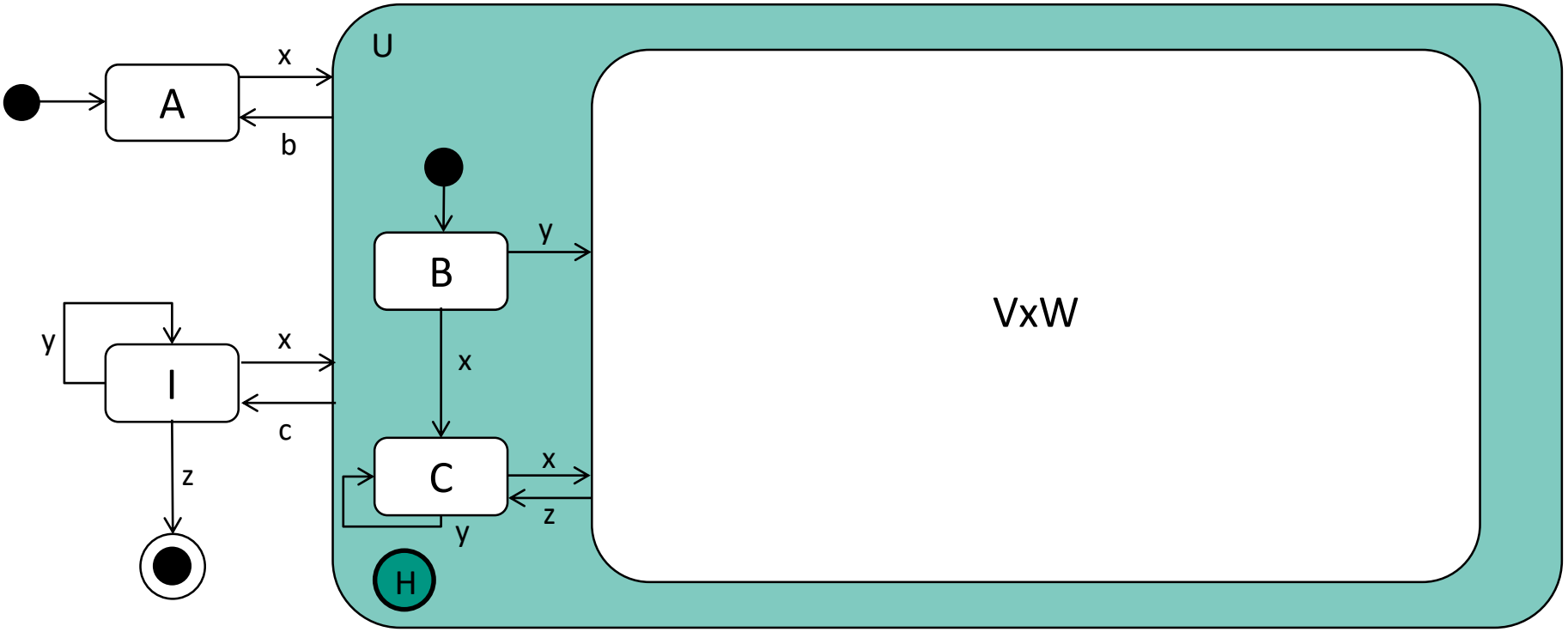
Aufgabe 5 – Zustandsdiagramm umwandeln



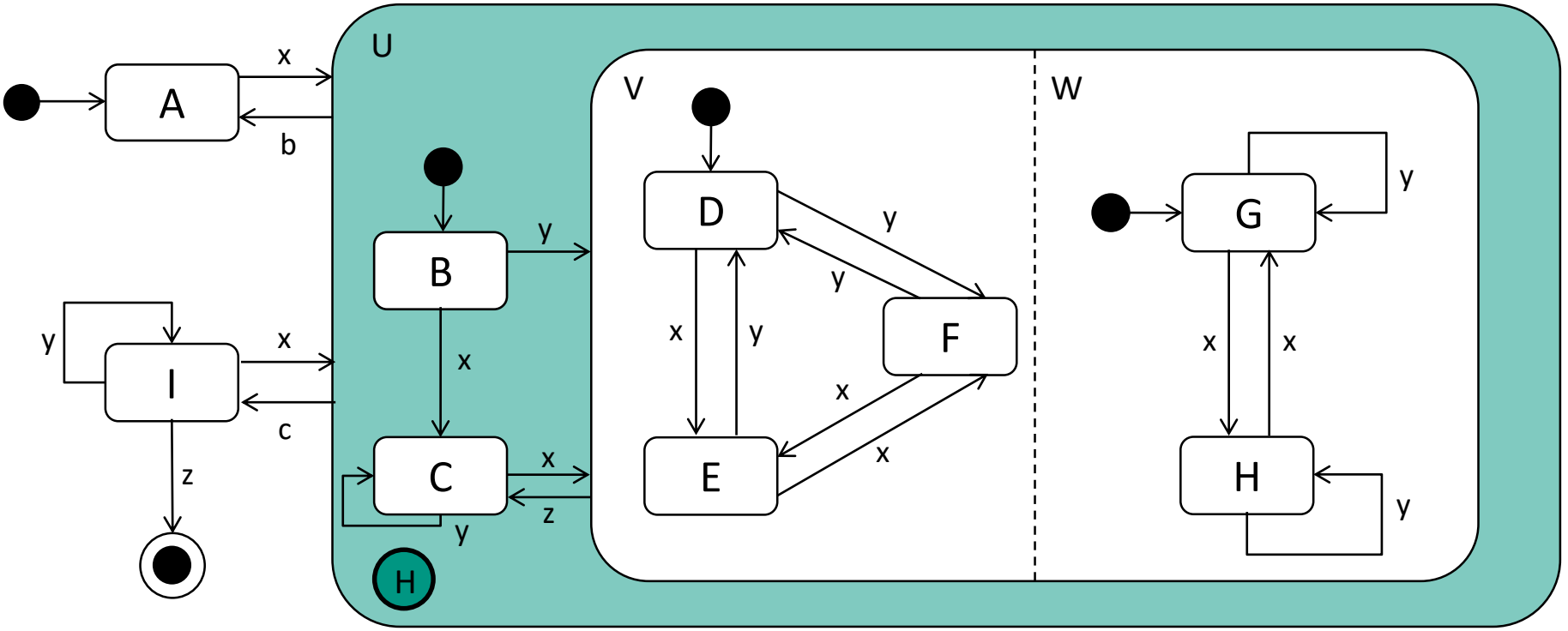
Aufgabe 5 – Zustandsdiagramm umwandeln



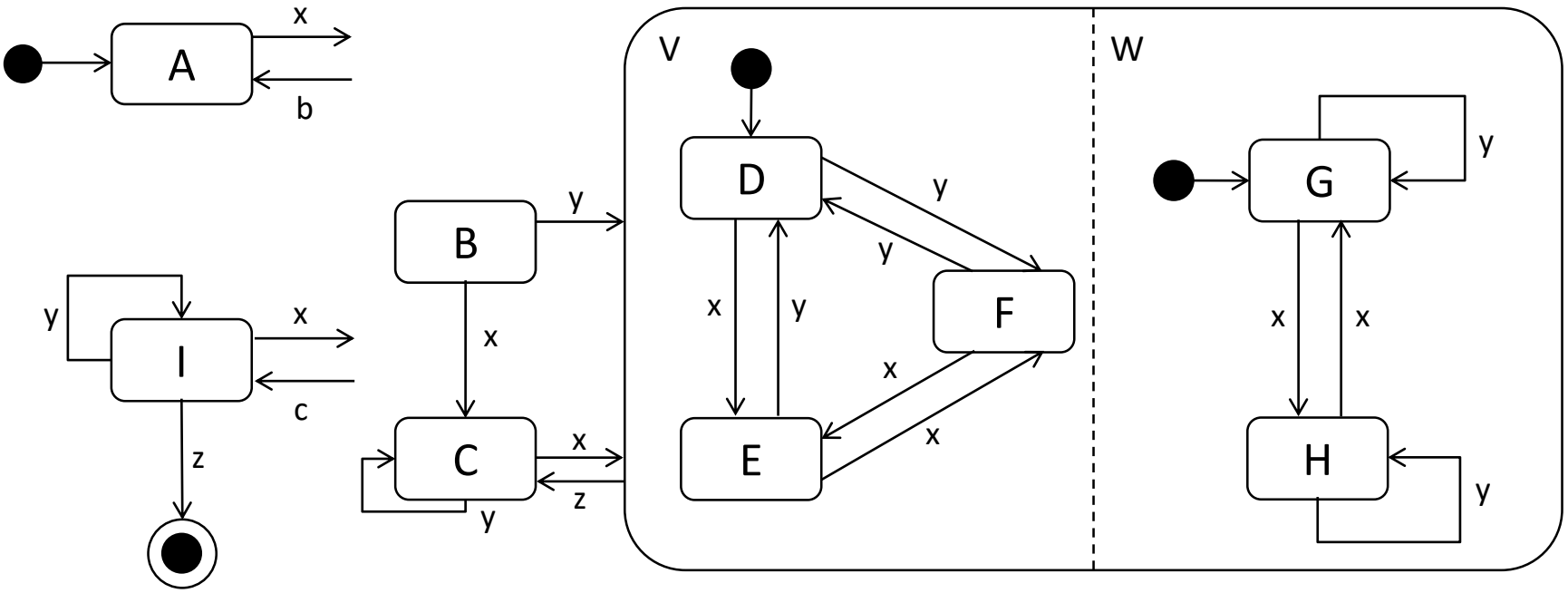
Aufgabe 5 – Zustandsdiagramm umwandeln



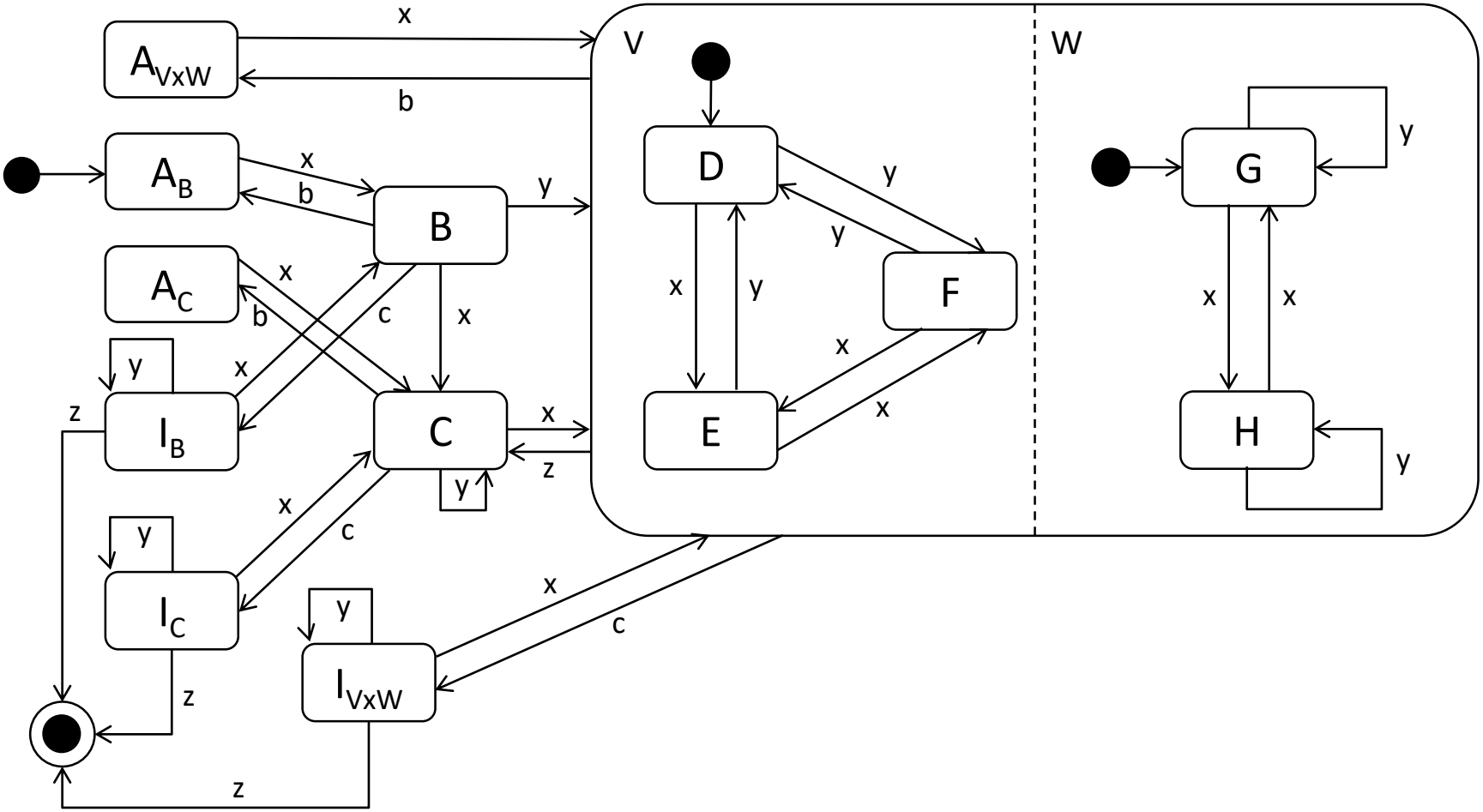
Aufgabe 5 – Zustandsdiagramm umwandeln



Aufgabe 5 – Zustandsdiagramm umwandeln



Aufgabe 5 – Zustandsdiagramm umwandeln



UML-Sequenzdiagramm

AUFGABE 6

Aufgabe 6 – Sequenzdiagramm zu HDrize

Die Kommandozeilen-Applikation (Klasse `App`*) ruft in der Klasse* `HDrize` *die* `createRGB`*-Methode (*`createRGB(EnhancedImage[] enhancedImages, int samples, double lambda, IMatrixCalculator<Matrix> mtxCalc, ToneMapping mapping)`*) auf. Diese Methode ruft wiederum die* `createHDR(…)`*-Methode (ebenfalls in* `HDrize`*) mit den entsprechenden Parametern auf. In* `createHDR(…)` *wird zunächst die Methode* `createCurve(…)` *(auch in* `HDrize`*) aufgerufen. Diese erzeugt eine neue Instanz der Klasse* `CameraCurve`*. Anschließend wird auf dieser Instanz die Methode* `calculate(…)` *mit den entsprechenden Parametern aufgerufen. Dann wird auf der gleichen Instanz* `isCalculated()` *aufgerufen, um zu prüfen, ob die Kurve berechnet wurde. Wurde die Kurve berechnet, wird die Instanz an* `createHDR(…)` *zurückgeliefert, andernfalls wird* `null` *zurückgeliefert.* `createHDR(…)` *ruft anschließend die Methode* `createHDR(…)` *in der Klasse* `HDRCombine` *auf. Diese liefert ein* `HDRImage` *zurück. Das erhaltene HDR-Bild wird anschließend an die* `createRGB(…)`*-Methode zurückgeliefert. Dort wird die statische Methode* `createRGB(…)` *in der Klasse* `HDRImageIO` *aufgerufen, die ein* `BufferedImage` *zurückliefert. Dieses wird von der Methode* `createRGB(…)` *in* `HDrize` *wiederum an die Kommandozeile zurückgeliefert.*

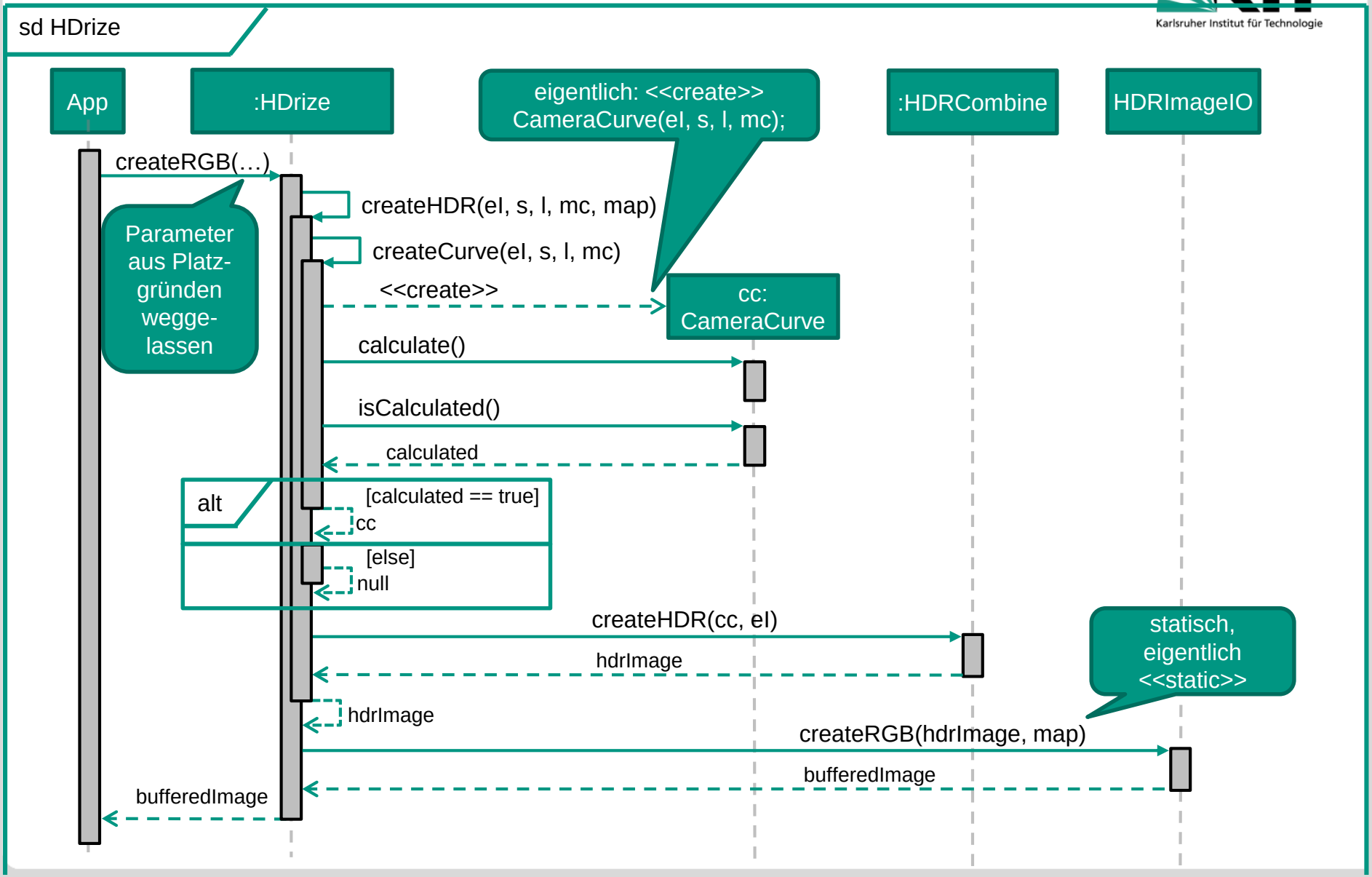
Aufgabe 6 – Sequenzdiagramm zu HDrize

```
public class HDrize implements IHDrize<Matrix> {
  // …
  @Override
  public BufferedImage createRGB(EnhancedImage[] enhancedImages,
    int samples, double lambda,
      IMatrixCalculator<Matrix> mtxCalc, ToneMapping mapping) {

    return HDRImageIO.createRGB(
        this.createHDR(enhancedImages, samples, lambda, mtxCalc),
        Obje      requireNonNull(mapping, "mapping cannot be null")
    );
  }
// …
}
```

erzeugt RGB-Variante des Bildes

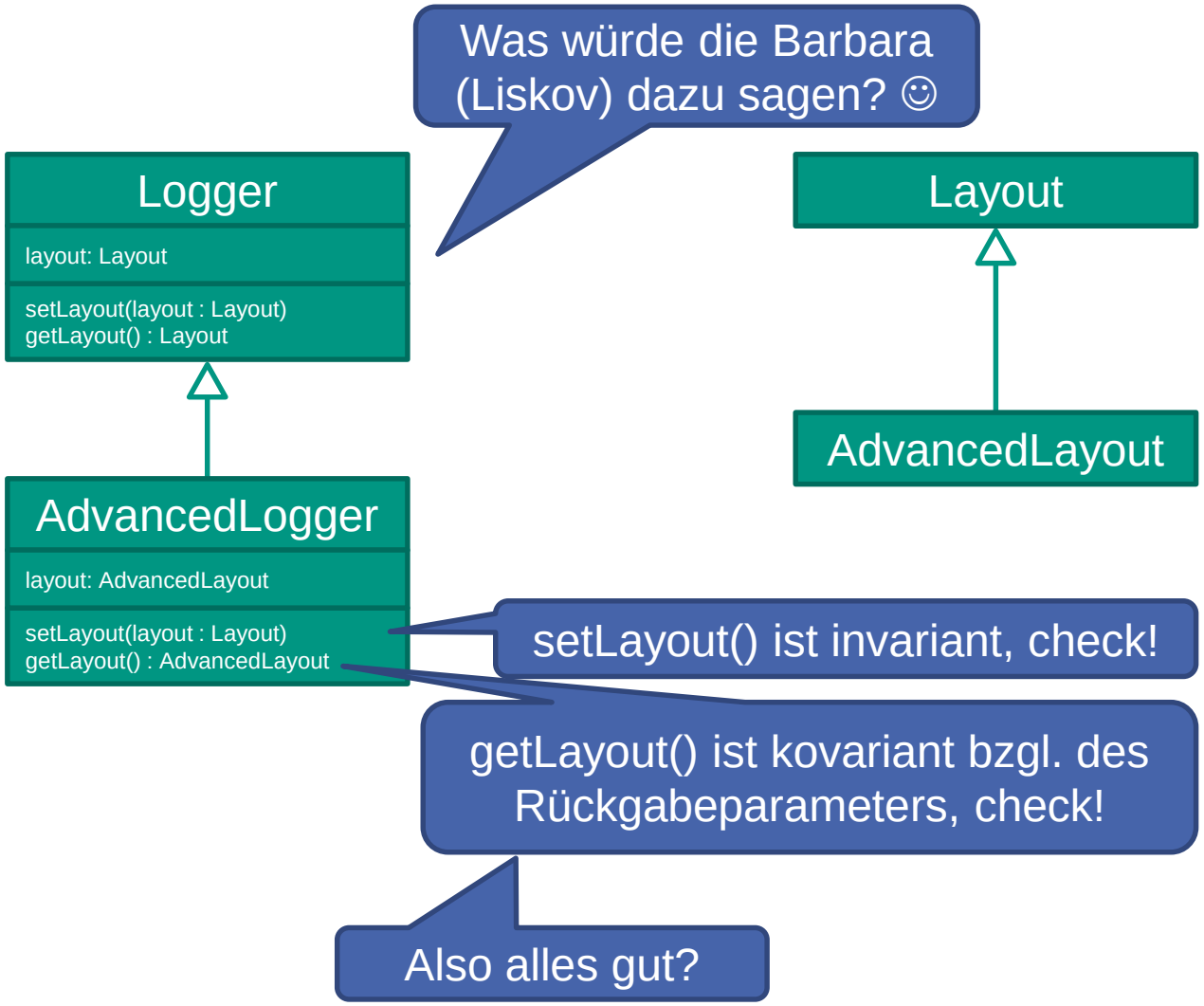
ruft wiederum createCurve(…) und HDRCombine. createHDR(…) auf


sd HDrize
App
:HDrize
eigentlich: <<create>>
CameraCurve(eI, s, l, mc);
:HDRCombine
HDRImageIO
createRGB(…)
Parameter aus Platz-gründen wegge-lassen
createHDR(eI, s, l, mc, map)
createCurve(eI, s, l, mc)
<<create>>
cc: CameraCurve
calculate()
isCalculated()
calculated
alt
[calculated == true]
cc
[else]
null
createHDR(cc, eI)
hdrImage
statisch, eigentlich <<static>>
hdrImage
createRGB(hdrImage, map)
bufferedImage
bufferedImage

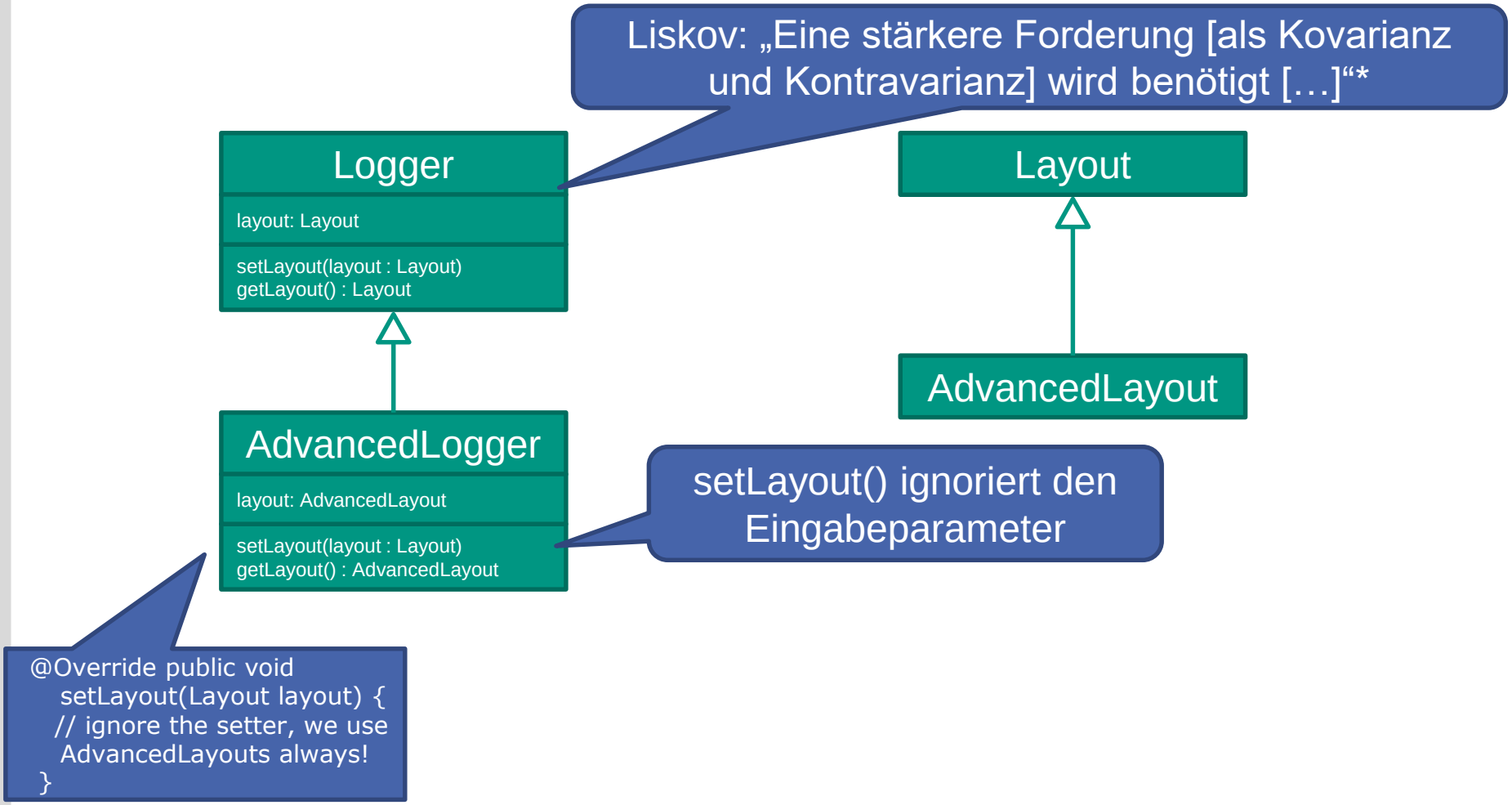
Augen auf bei der Bibliothekswahl

AUFGABE 7

Aufgabe 7 – Augen auf bei der Bibliothekswahl

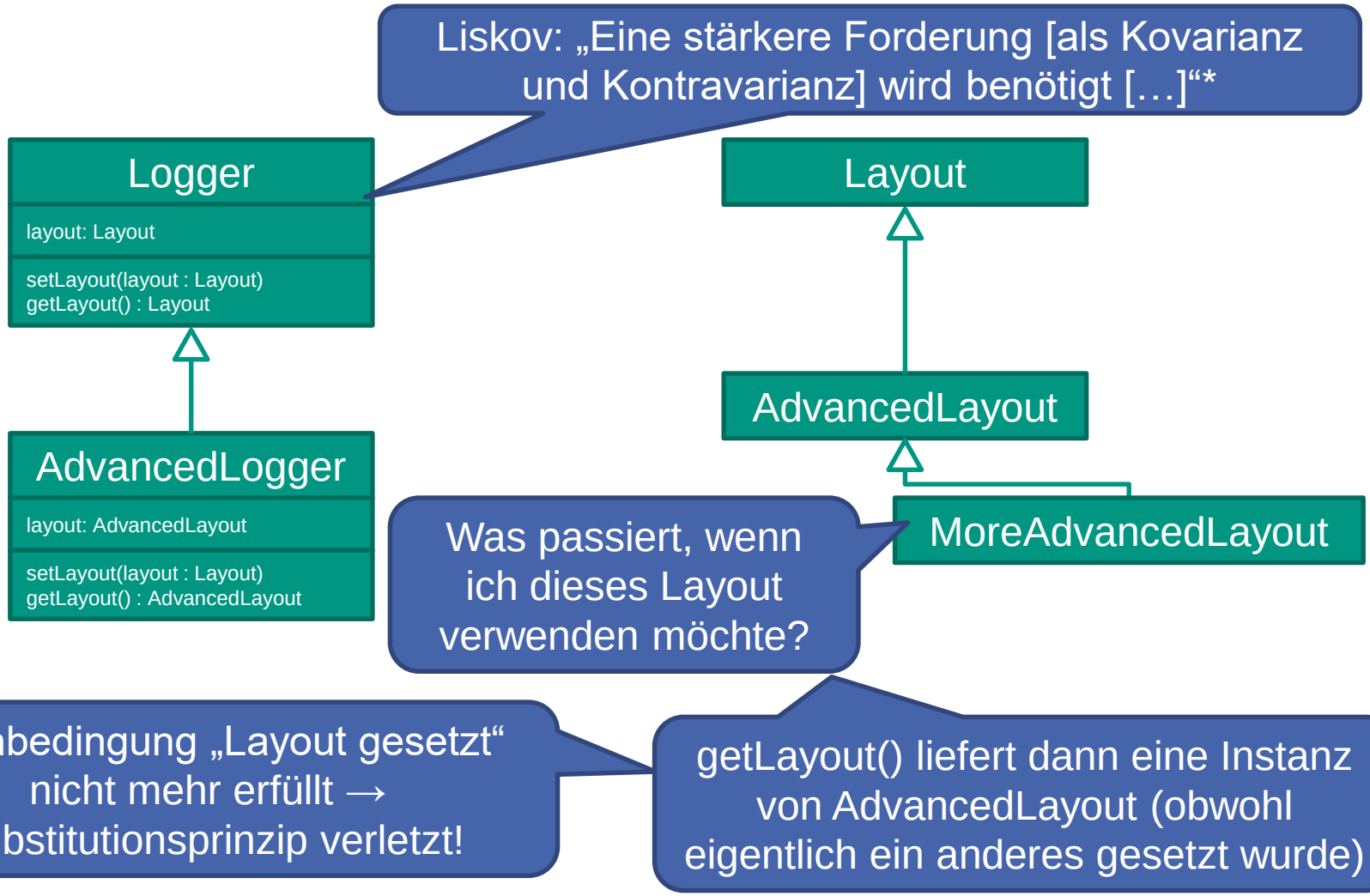


Aufgabe 7 – Augen auf bei der Bibliothekswahl



*Liskov, Barbara ; Wing, Jeannette M: Family Values: A Behavioral Notion of Subtyping, Pittsburgh 1993

Aufgabe 7 – Augen auf bei der Bibliothekswahl



*Liskov, Barbara ; Wing, Jeannette M: Family Values: A Behavioral Notion of Subtyping, Pittsburgh 1993

Aufgabe 7 – Augen auf bei der Bibliothekswahl

- Moral:
  - Das Liskov‘sche Substitutionsprinzip ist mehr als nur die Betrachtung der Varianz der Ein- und Ausgabeparameter
  - Zulässige Varianzen stellen lediglich eine notwendige, jedoch keine hinreichende Bedingung dar
  - Prüfe daher Vor- und Nachbedingungen…
  - … oder einfach *„Verhält sich jedes Exemplar der Unterklasse in allen Kontexten wie ein Exemplar der Oberklasse?“*
  - Derartige Probleme treten im wahren Leben auf*

* Aufgabe angelehnt an: https://devonblog.com/software-development/solid-violations-wild-liskov-substitution-principle/