AVB for low latency / industrial networks:

Media redundancy for fault tolerance and AVB - continuation

Oliver Kleineberg, Hirschmann Automation & Control
(oliver.kleineberg@belden.com)

IEEE 802 Plenary Meeting, March 2011, Singapore

Media redundancy and AVB

- **Aims of this Presentation:**
- **To take the idea of media redundancy for fault tolerance and AVB one step further and give some perspective on „how-to“**
- **These first solution proposals are (of course) raw and unpolished, but show that media redundancy and AVB can work together**
- **Proposed solutions define a common ground for everybody to start thinking further**

Agenda

- Short flashback to Dallas Meeting:
  - Conceptual approach to media redundancy
  - Why configured VLANs are not a feasible solution
- Further insight into possibilities of realization:
  - Redundant path registration
  - Support for fault-tolerant networks with and without communication interruption
- From arbitrarily meshed networks to selected paths

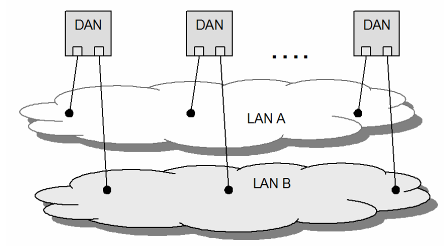
Flashback to Dallas meeting

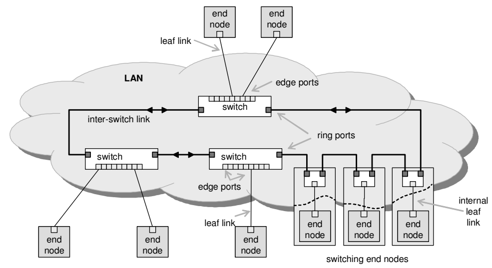
Short flashback to Dallas meeting –

Conceptual approach to media redundancy

Flashback to Dallas meeting

	redundant links	redundant networks
with network interruption		
without network interruption		





- In theory, all four combinations are possible
- In practice, some configurations are far more widely used than others, but all possibilities need to be covered by a solution

- (if possible) full coverage of all combinations with as few mechanisms as possible
- Mechanisms should not require extensive manual configuration
- Manual configuration is acceptable (and sometimes desired) to some extent in „engineered networks“

Flashback to Dallas meeting

Short flashback to Dallas meeting –

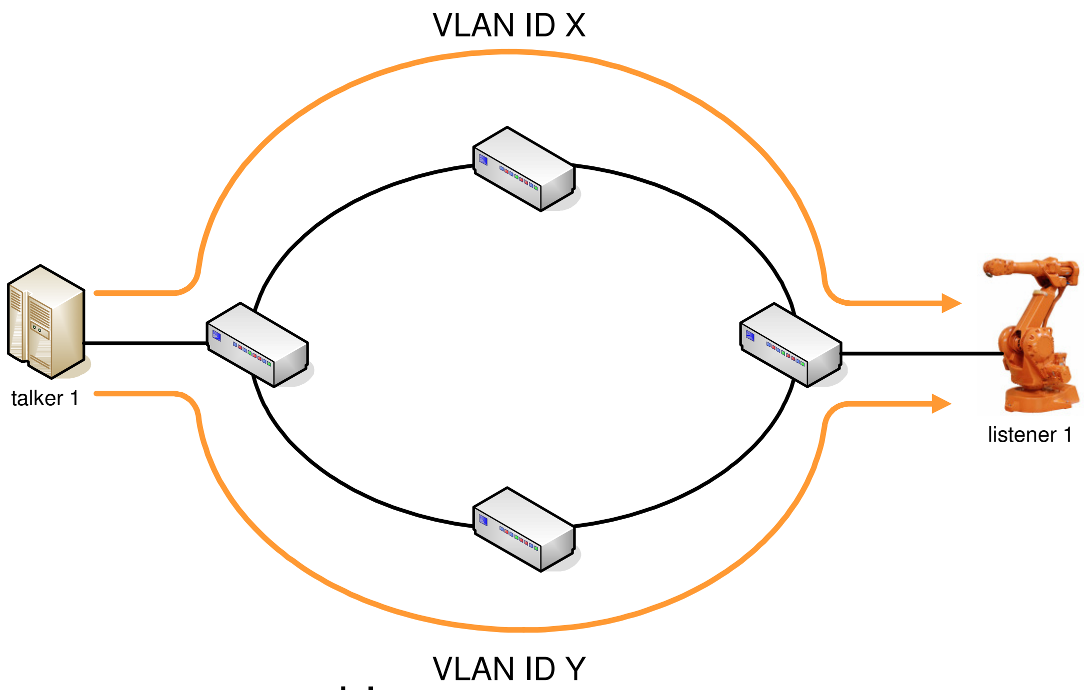
Why VLANs are not a feasible solution

VLANs – not the instrument of choice for redundant paths

Application example from Dallas presentation: Proposed solution by using different VLANs on two distinct physical paths

Problem 1:

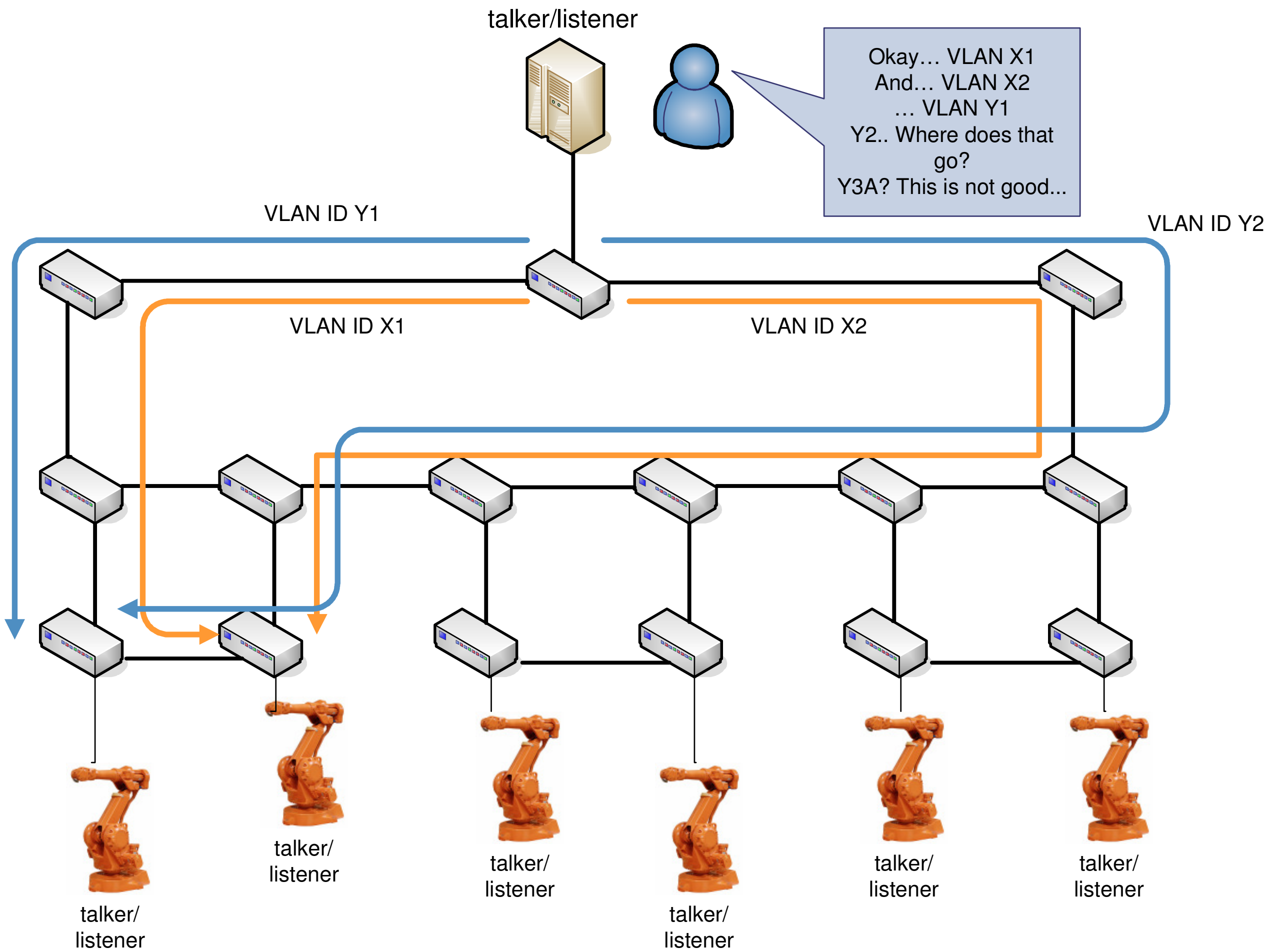
→ **Not configuration free, possibly high configuration effort**



→ Simple structures are manageable

VLANs – not the instrument of choice for redundant paths

Problem 1: → more complex topologies will quickly overwhelm users (even with SCADA support)

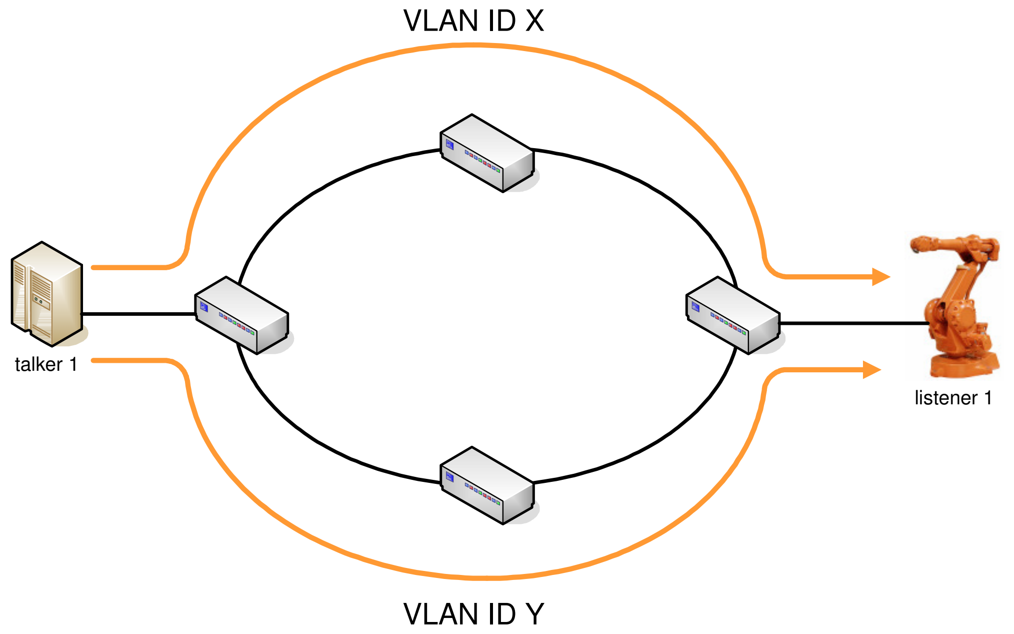


VLANs – not the instrument of choice for redundant paths

Application example from Dallas presentation: Proposed solution by using different VLANs on two distinct physical paths

Problem 2:

→ Blocking of (several) VLAN IDs for application purposes and the challenge of distinguishing between VLANs for applications and VLANs for redundancy: Makes network management error prone and complicated



Result: (Manually) configured VLANs according to the physical redundant topology are not the instrument of choice to realize redundant streams!

We need another **idea…**

How can it be done?

Further insight into possibilities of realization -

Bridges establishing redundant streams

Bridges establishing redundant streams

Idea: „Mark“ streams that are meant to be sent redundantly and let bridges handle them accordingly

• Streams that are intended to be sent redundantly can be identified by a „redundancy identifier“ (to be defined, could be e.g. an attribute declaration) → Bridges track redundant streams by their ID and the redundancy identifier

• This „redundancy identifier“ can be either set by talkers that want a redundant network structure to handle its stream redundantly (or that have redundant network interfaces themselves) or it can be set by a bridge (e.g. a bridge that implements a redundancy protocol and that has a redundancy-unaware talker on one of its ports)

• Bridges produce (and consume) redundant stream registrations

Additional protocol information needed

•With the redundancy identifier, standard MSRP streams can be distinguished from redundant streams and can be handled accordingly by bridges that are „redundancy aware“

• MSRPDUs, or respectively the attributes, that have the redundancy identifier „set“ are transmitted over discarding ports. → Streams are registered on the path with the discarding port, but stream frames are not transmitted

→ discarding ports are effectively ignored for stream registration, but not for the actual stream transmission

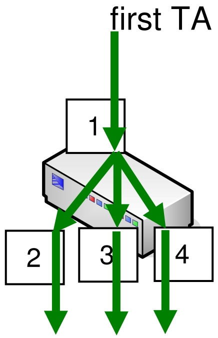
Part 1: talker advertise

Bridge behaviour:

• A bridge that receives a talker advertise and can identify the corresponding stream as redundant sends the advertisement to all ports it has not sent the advertisement (except the receiving port)

• If a bridge has sent the talker advertise to all ports, it drops all further talker advertise frames for that particular stream ID (until the leave-all interval has passed)

• A bridge registers on which ports it received the talker advertise



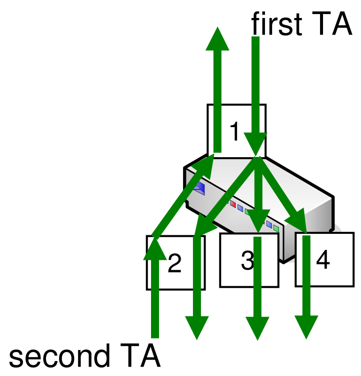
Part 1: talker advertise

Bridge behaviour:

• A bridge that receives a talker advertise and can identify the corresponding stream as redundant sends the advertisement to all ports it has not sent the advertisement (except the receiving port)

• If a bridge has sent the talker advertise to all ports, it drops all further talker advertise frames for that particular stream ID (until the leave-all interval has passed)

• A bridge registers on which ports it received the talker advertise



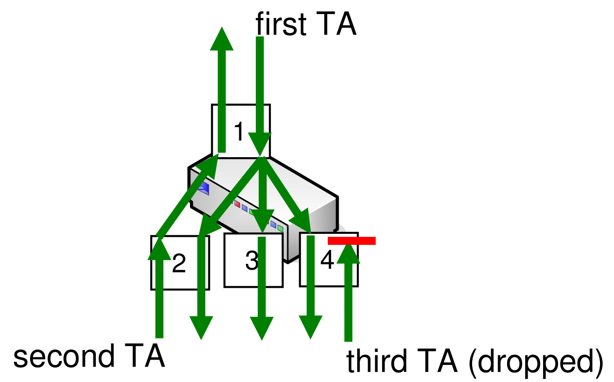
Part 1: talker advertise

Bridge behaviour:

• A bridge that receives a talker advertise and can identify the corresponding stream as redundant sends the advertisement to all ports it has not sent the advertisement (except the receiving port)

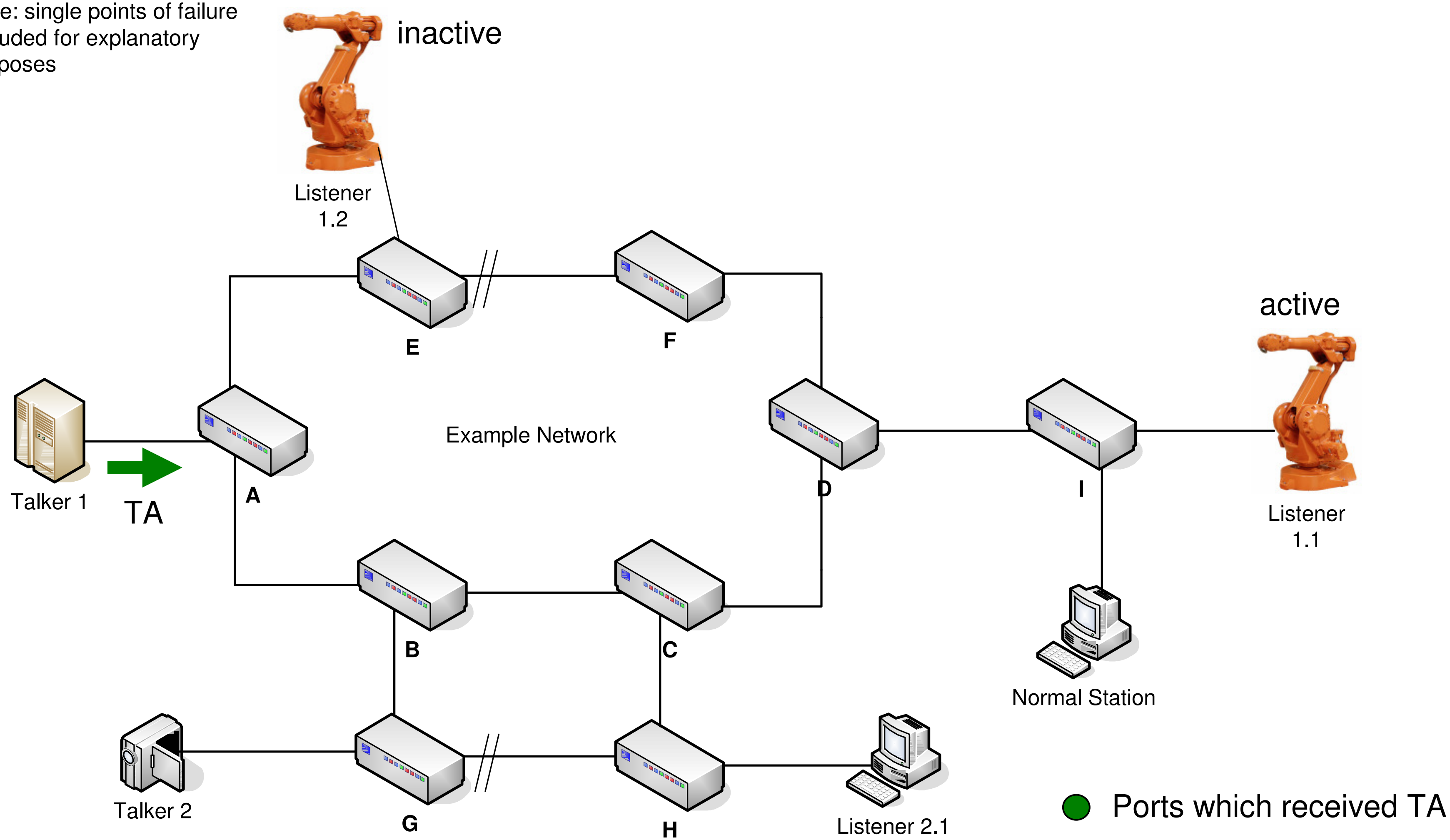
• If a bridge has sent the talker advertise to all ports, it drops all further talker advertise frames for that particular stream ID (until the leave-all interval has passed)

• A bridge registers on which ports it received the talker advertise



Example network - part 1: talker advertise

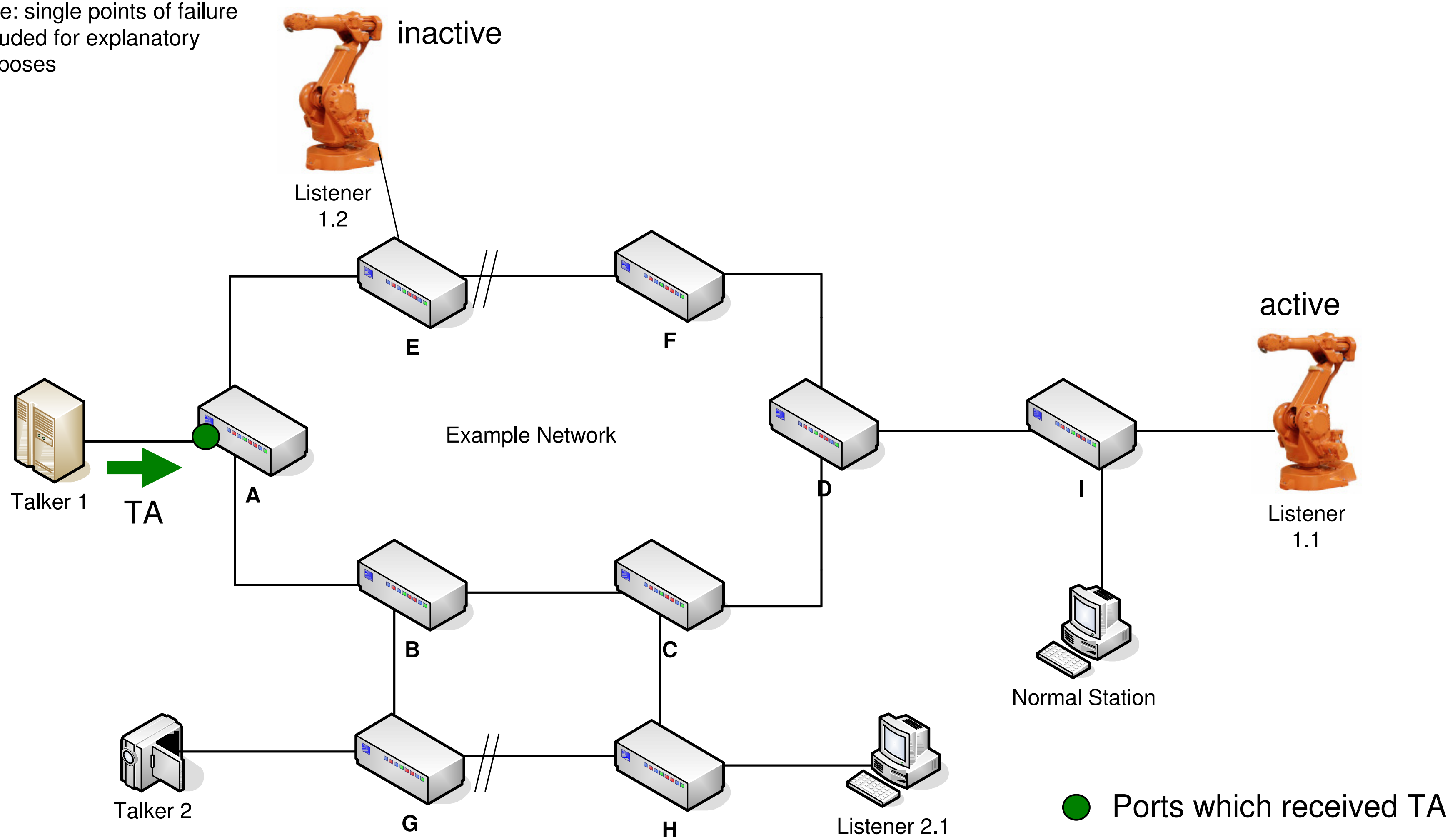
Note: single points of failure included for explanatory purposes



Talker advertisements are sent by bridges on all ports except the ports the same advertisement has been sent to already (and on which the same TA has been received)

Example network - part 1: talker advertise

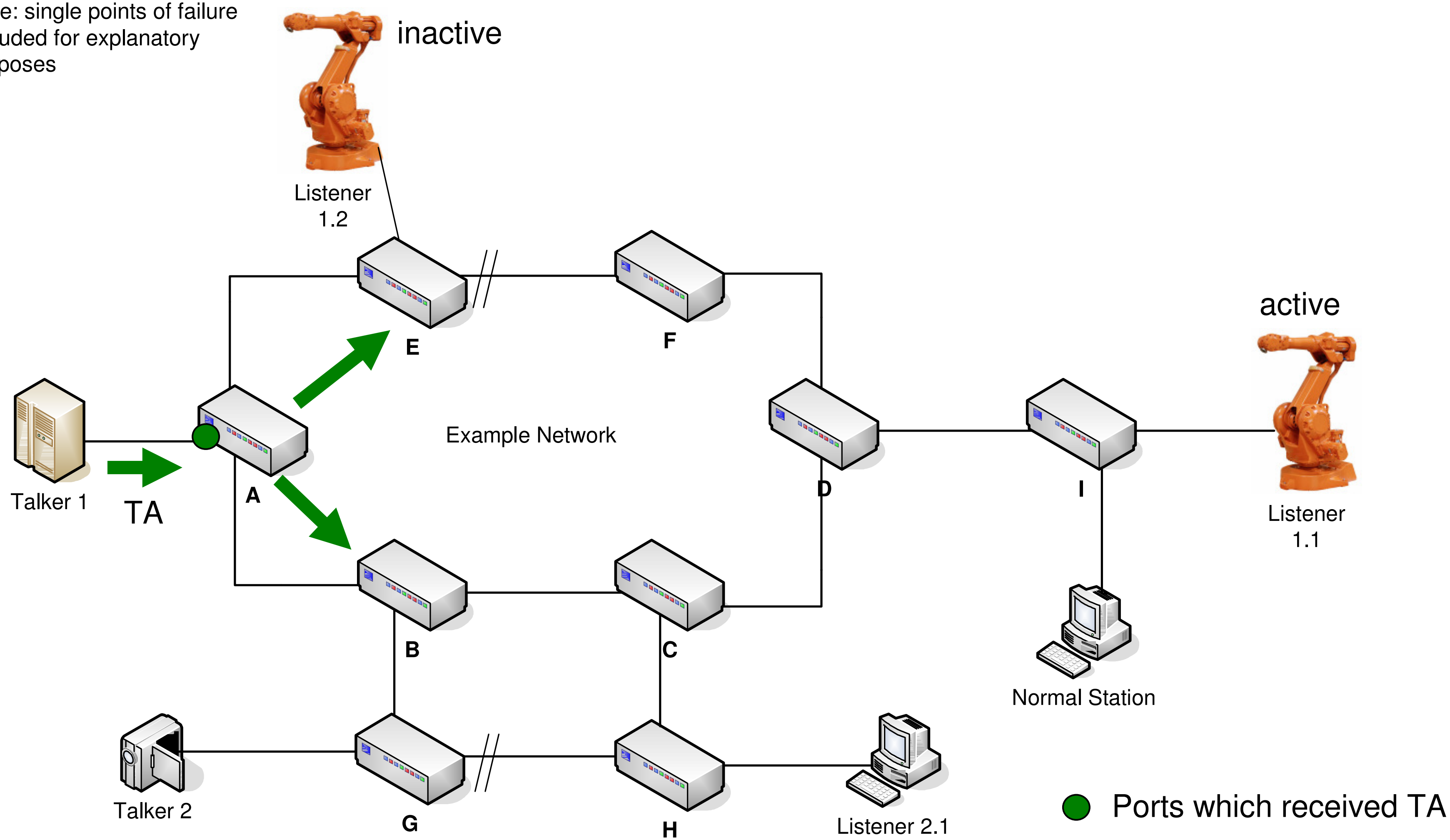
Note: single points of failure included for explanatory purposes



Talker advertisements are sent by bridges on all ports except the ports the same advertisement has been sent to already (and on which the same TA has been received)

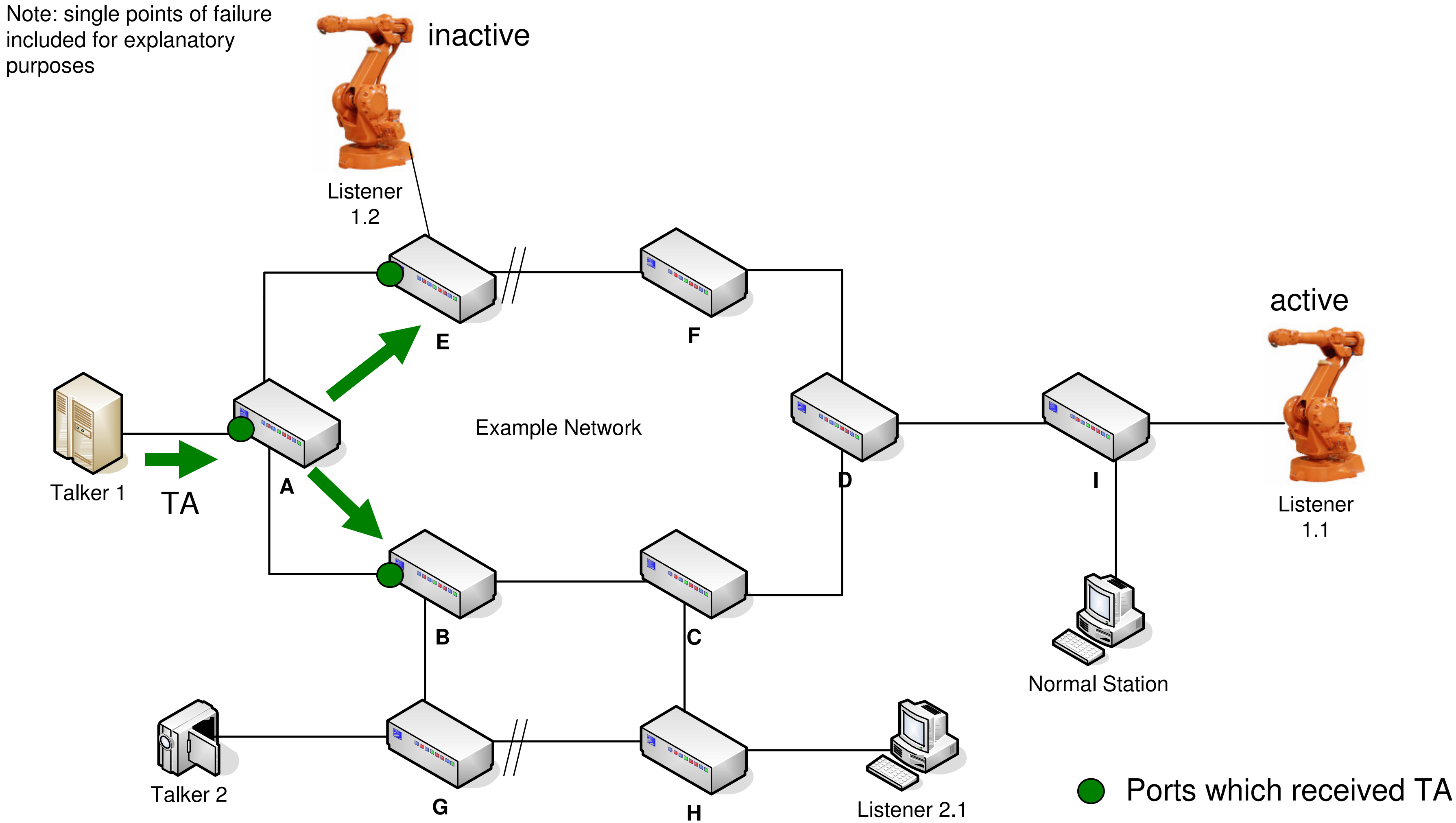
Example network - part 1: talker advertise

Note: single points of failure included for explanatory purposes



Talker advertisements are sent by bridges on all ports except the ports the same advertisement has been sent to already (and on which the same TA has been received)

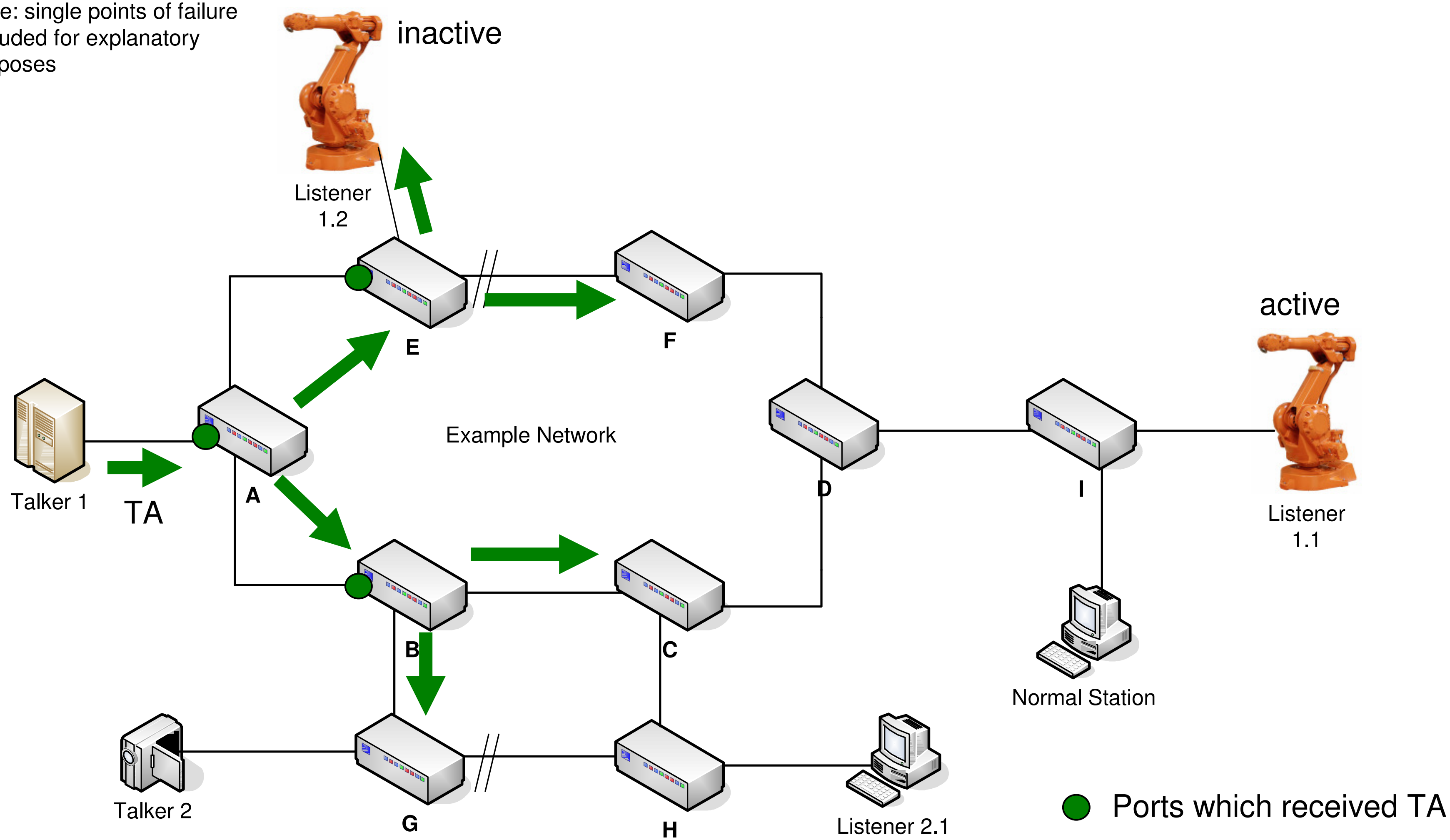
Example network - part 1: talker advertise



Talker advertisements are sent by bridges on all ports except the ports the same advertisement has been sent to already (and on which the same TA has been received)

Example network - part 1: talker advertise

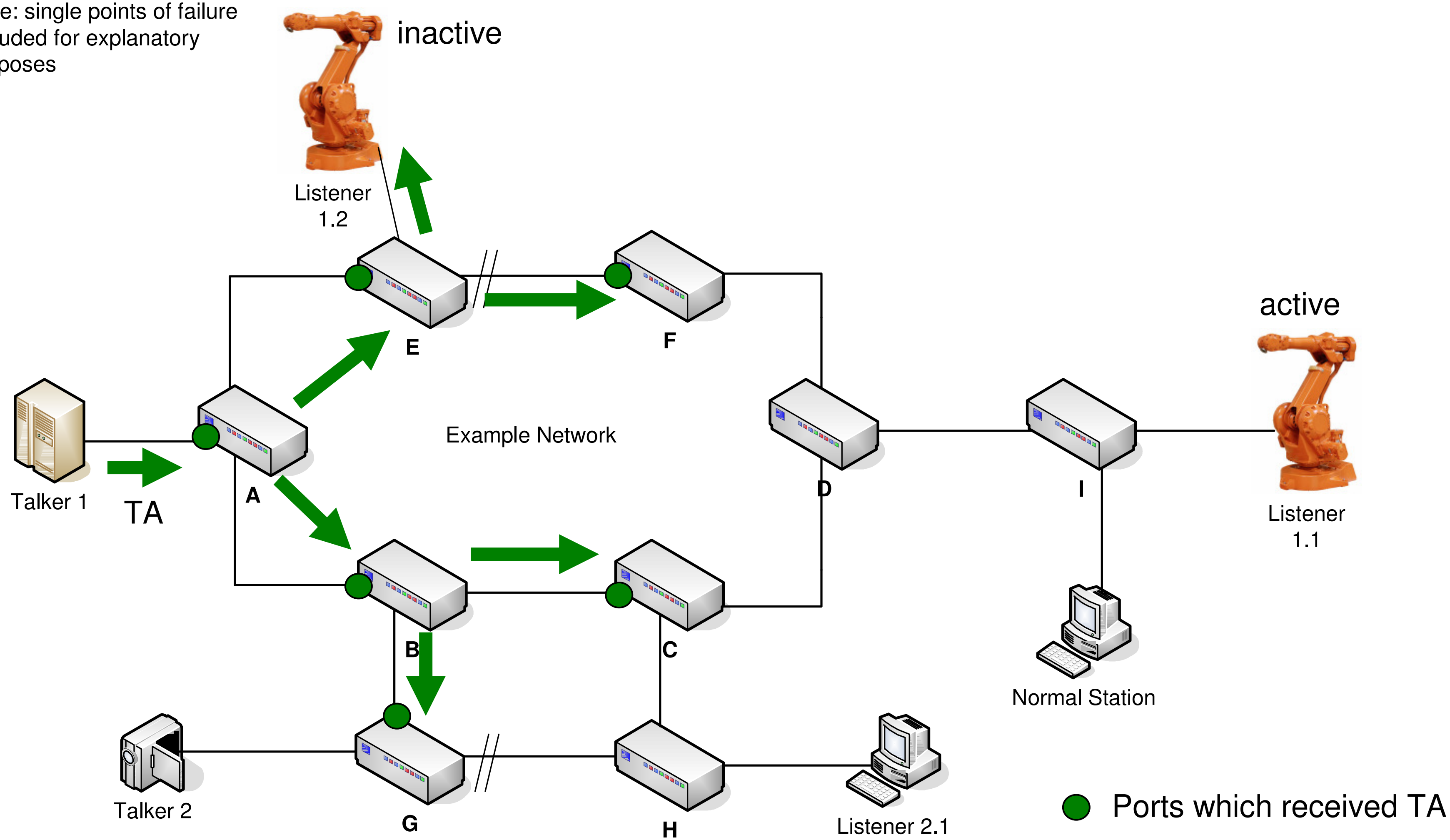
Note: single points of failure included for explanatory purposes



Talker advertisements are sent by bridges on all ports except the ports the same advertisement has been sent to already (and on which the same TA has been received)

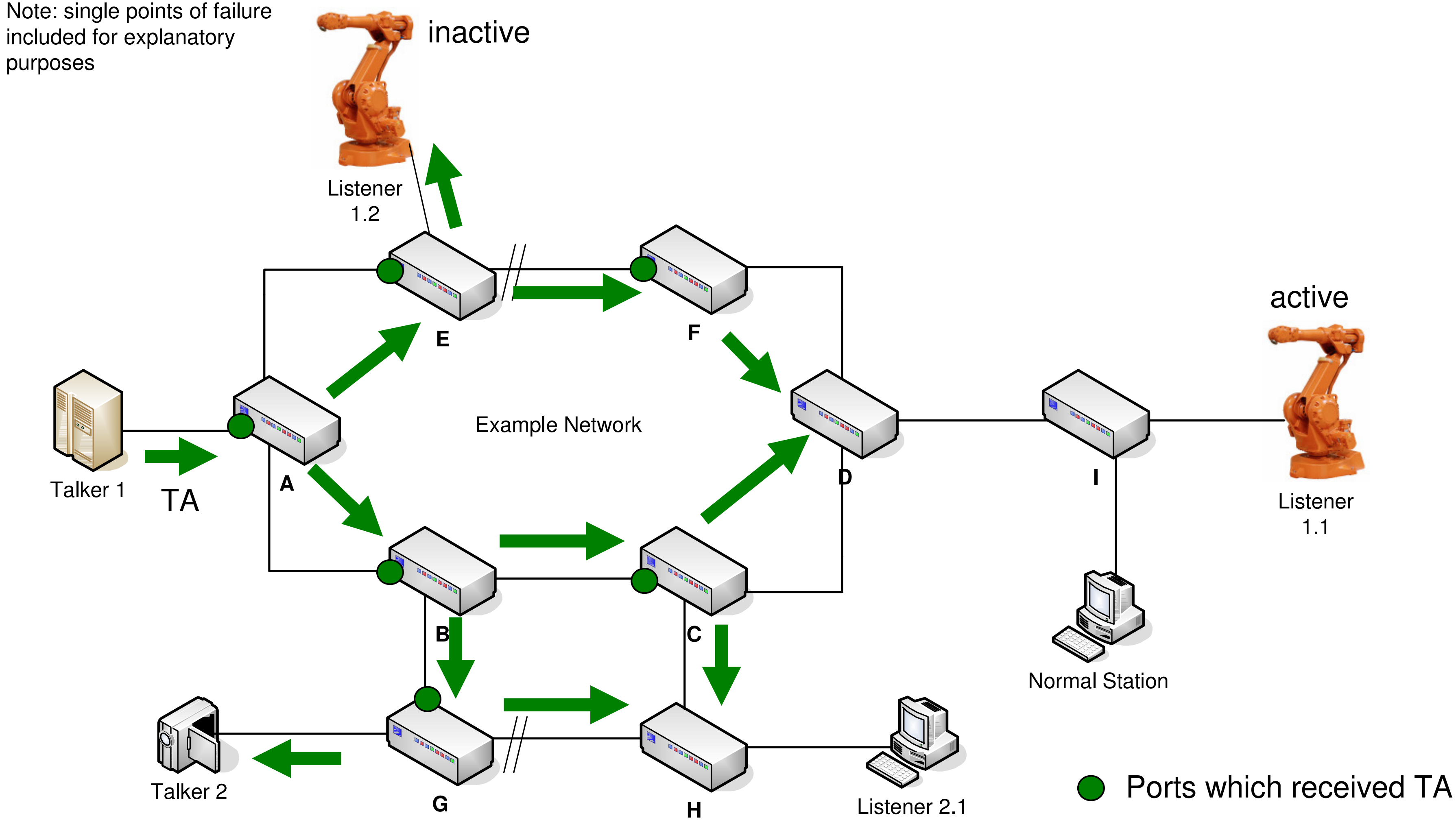
Example network - part 1: talker advertise

Note: single points of failure included for explanatory purposes



Talker advertisements are sent by bridges on all ports except the ports the same advertisement has been sent to already (and on which the same TA has been received)

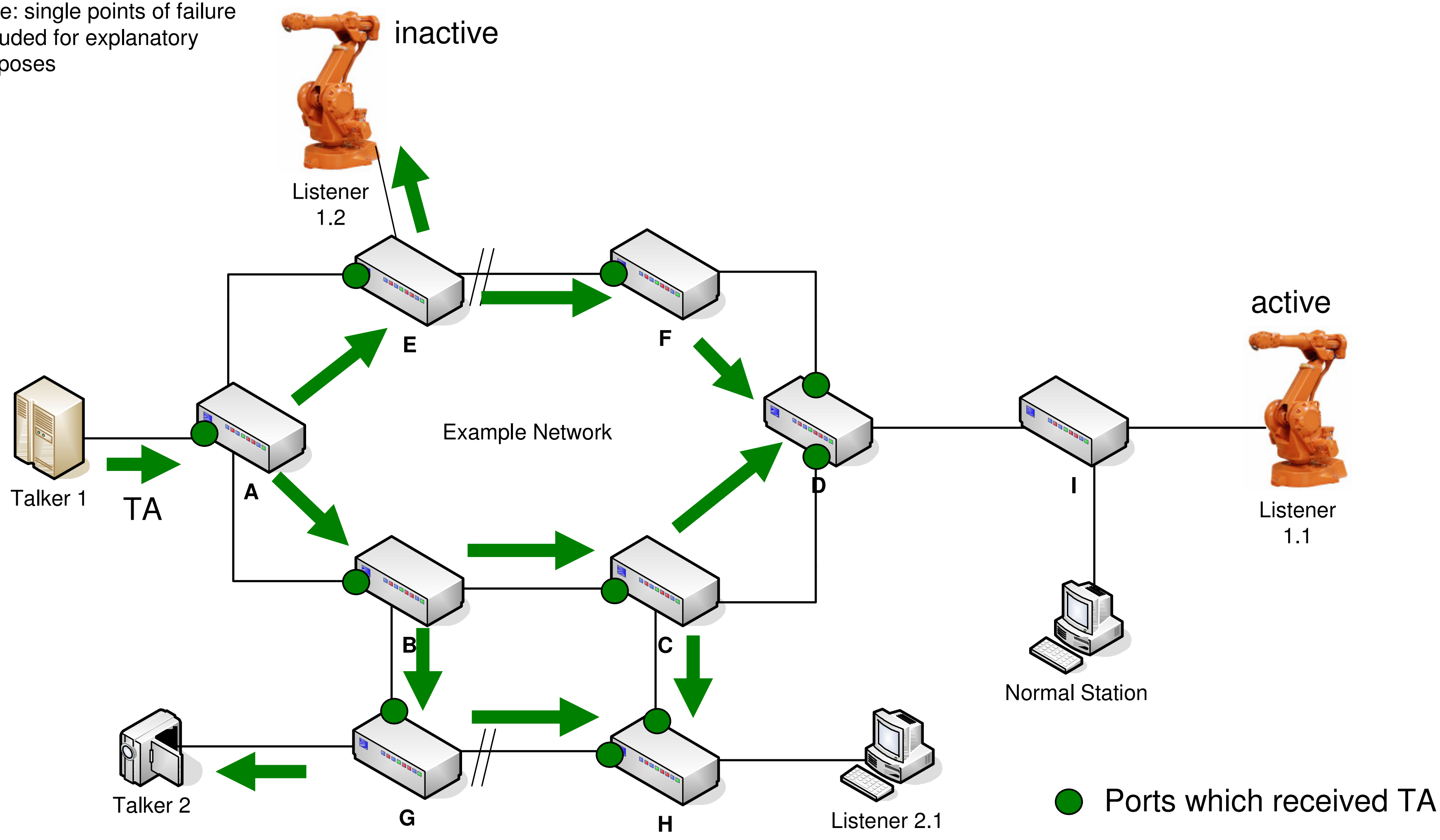
Example network - part 1: talker advertise



Talker advertisements are sent by bridges on all ports except the ports the same advertisement has been sent to already (and on which the same TA has been received)

Example network - part 1: talker advertise

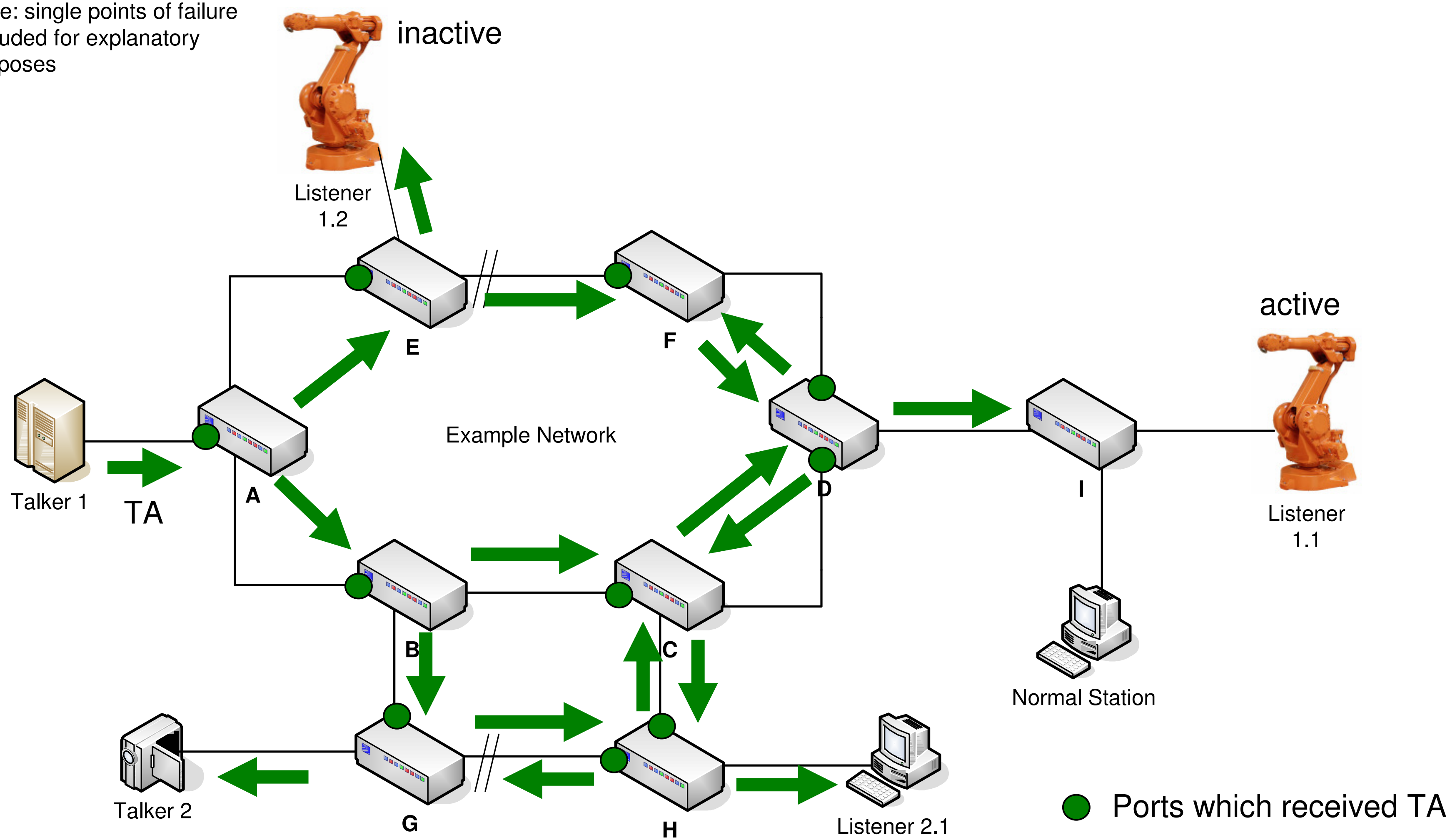
Note: single points of failure included for explanatory purposes



Talker advertisements are sent by bridges on all ports except the ports the same advertisement has been sent to already (and on which the same TA has been received)

Example network - part 1: talker advertise

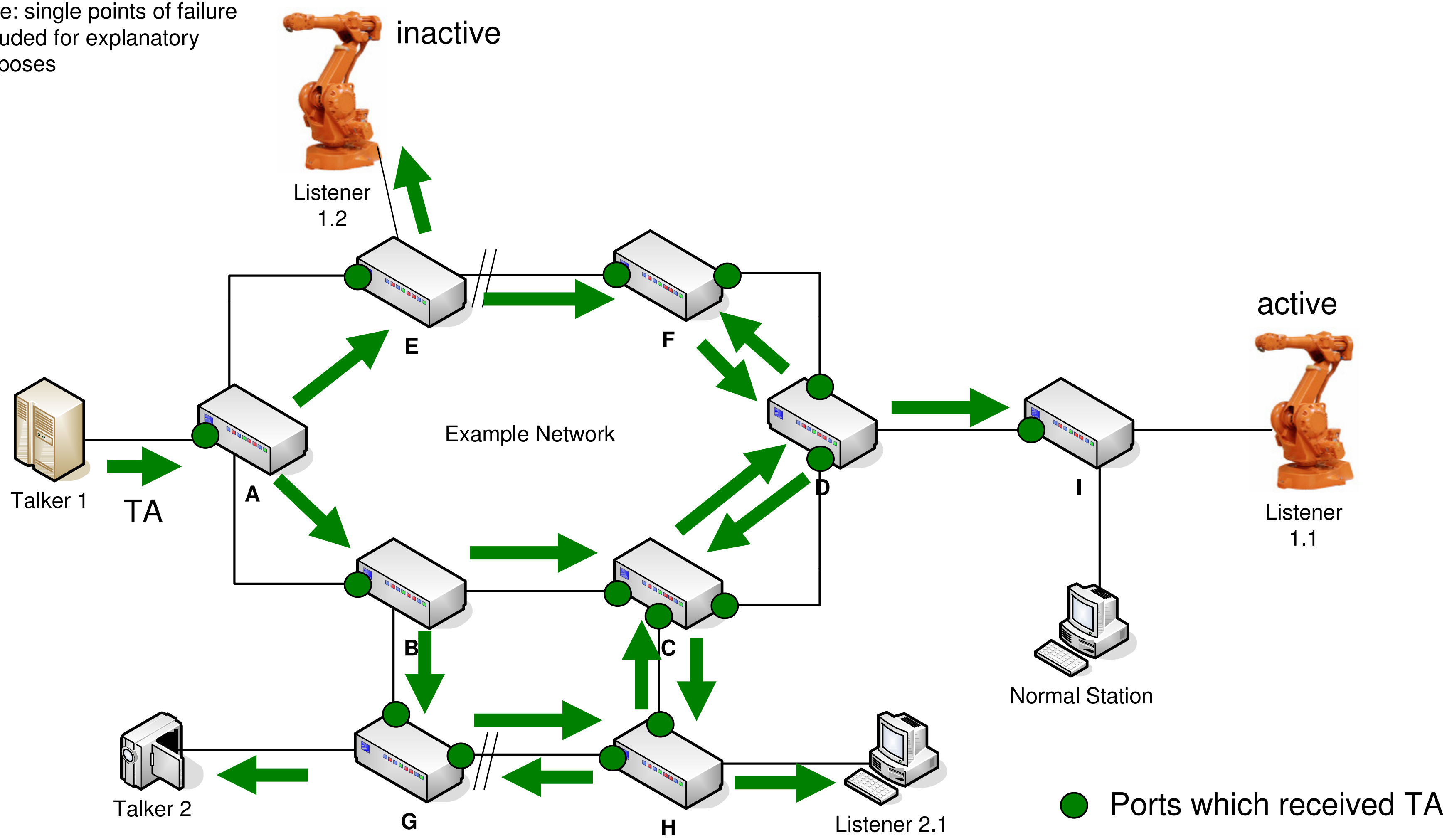
Note: single points of failure included for explanatory purposes



Talker advertisements are sent by bridges on all ports except the ports the same advertisement has been sent to already (and on which the same TA has been received)

Example network - part 1: talker advertise

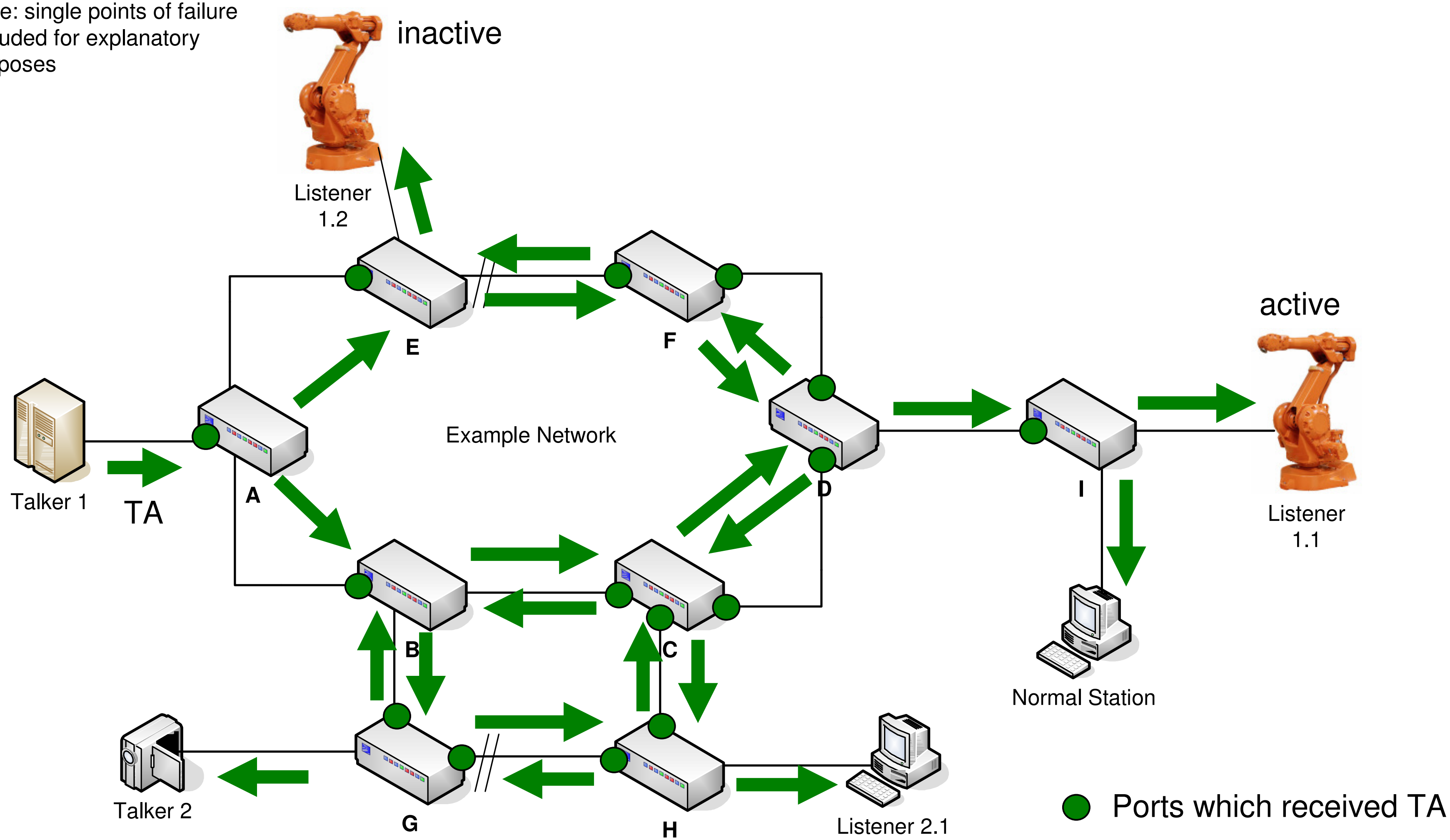
Note: single points of failure included for explanatory purposes



Talker advertisements are sent by bridges on all ports except the ports the same advertisement has been sent to already (and on which the same TA has been received)

Example network - part 1: talker advertise

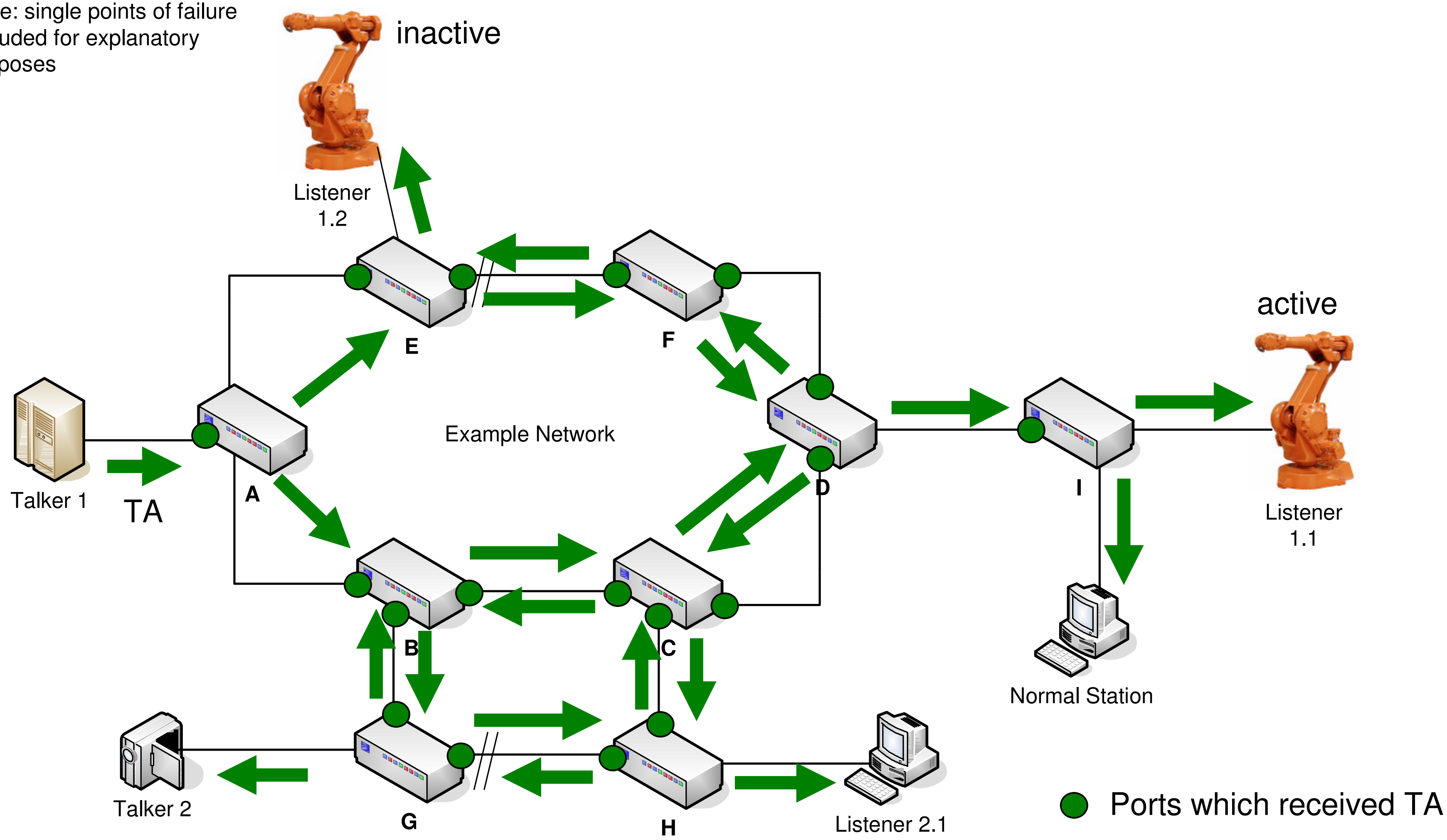
Note: single points of failure included for explanatory purposes



Talker advertisements are sent by bridges on all ports except the ports the same advertisement has been sent to already (and on which the same TA has been received)

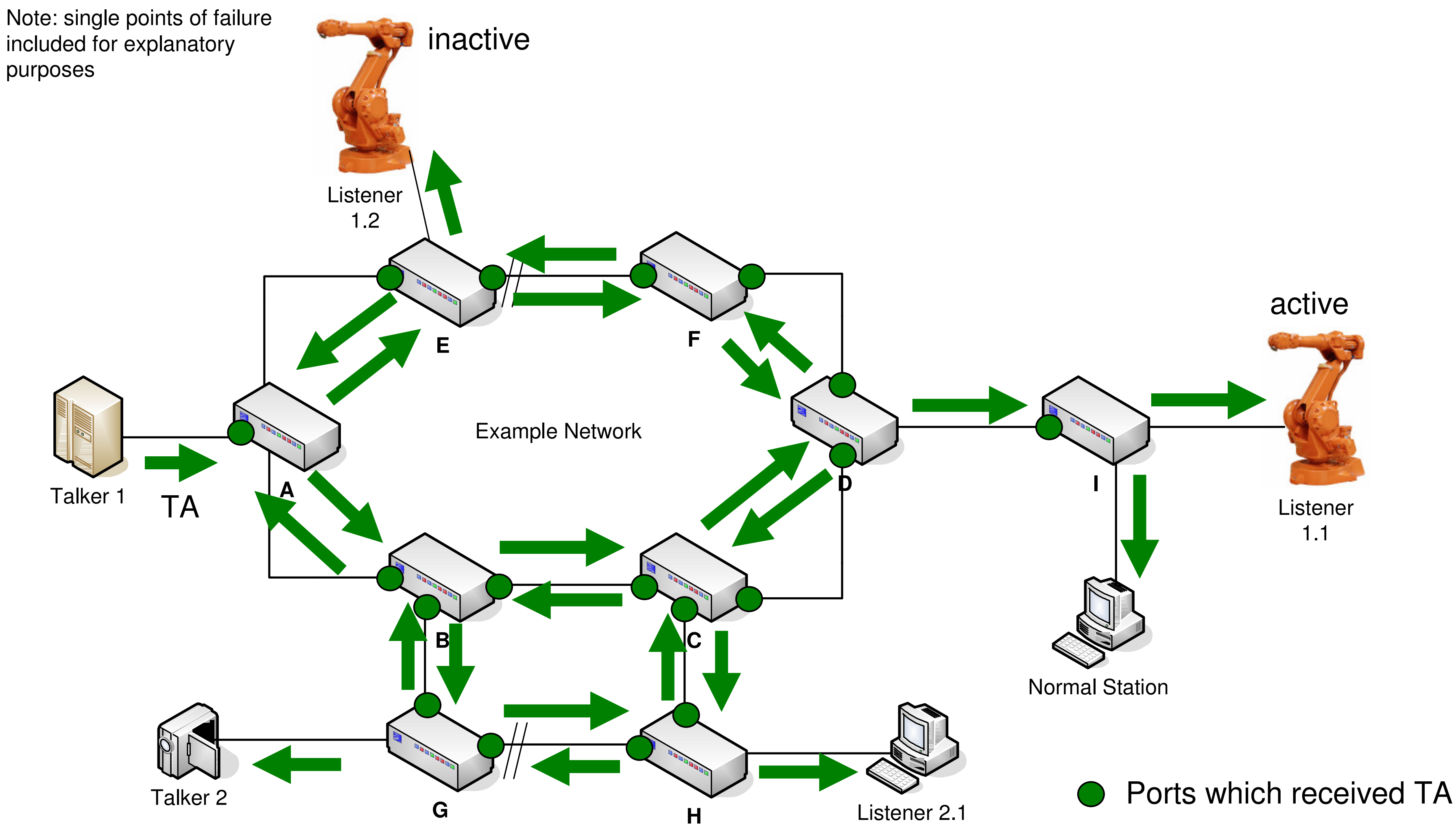
Example network - part 1: talker advertise

Note: single points of failure included for explanatory purposes



Talker advertisements are sent by bridges on all ports except the ports the same advertisement has been sent to already (and on which the same TA has been received)

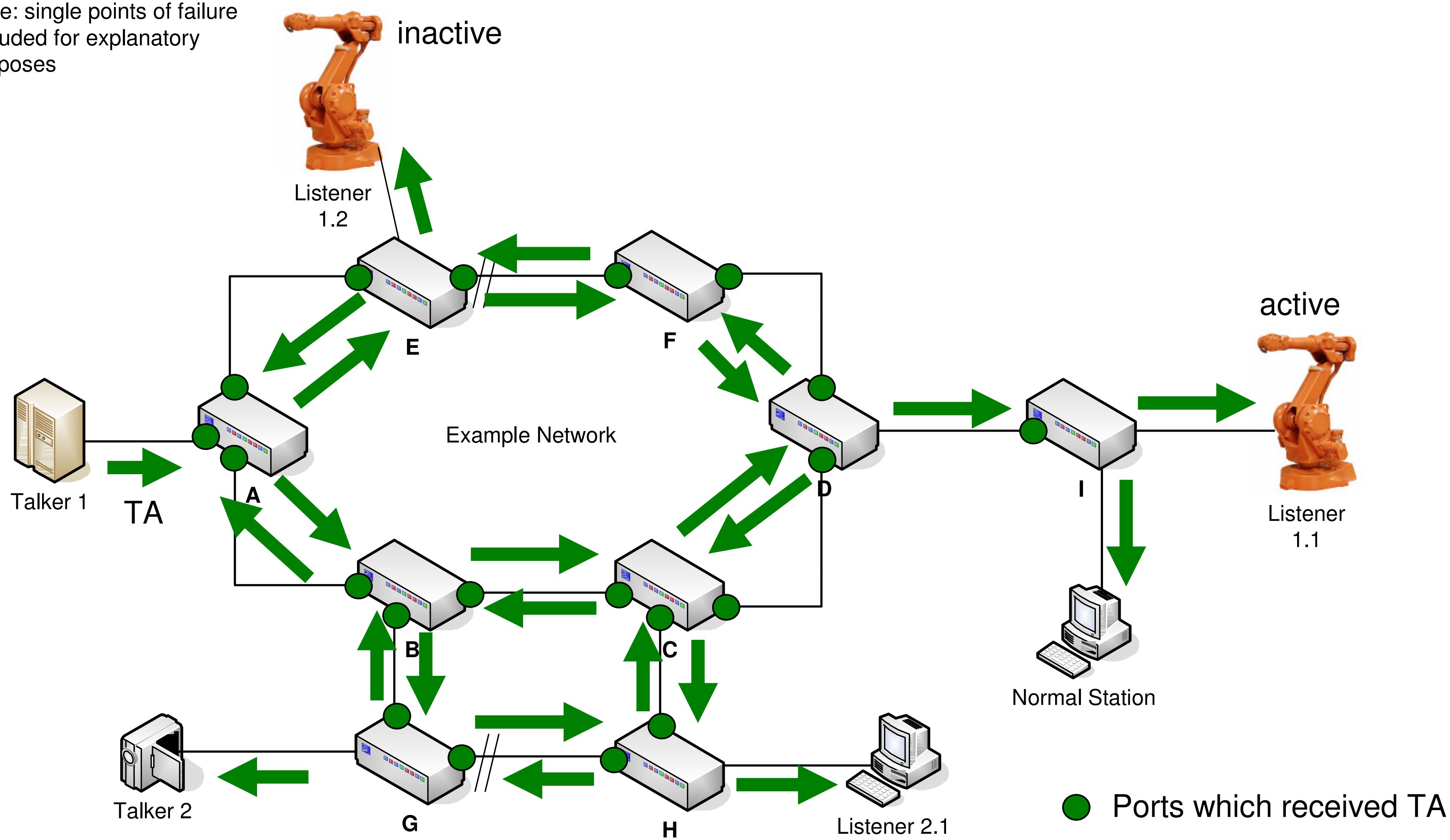
Example network - part 1: talker advertise



Talker advertisements are sent by bridges on all ports except the ports the same advertisement has been sent to already (and on which the same TA has been received)

Example network - part 1: talker advertise

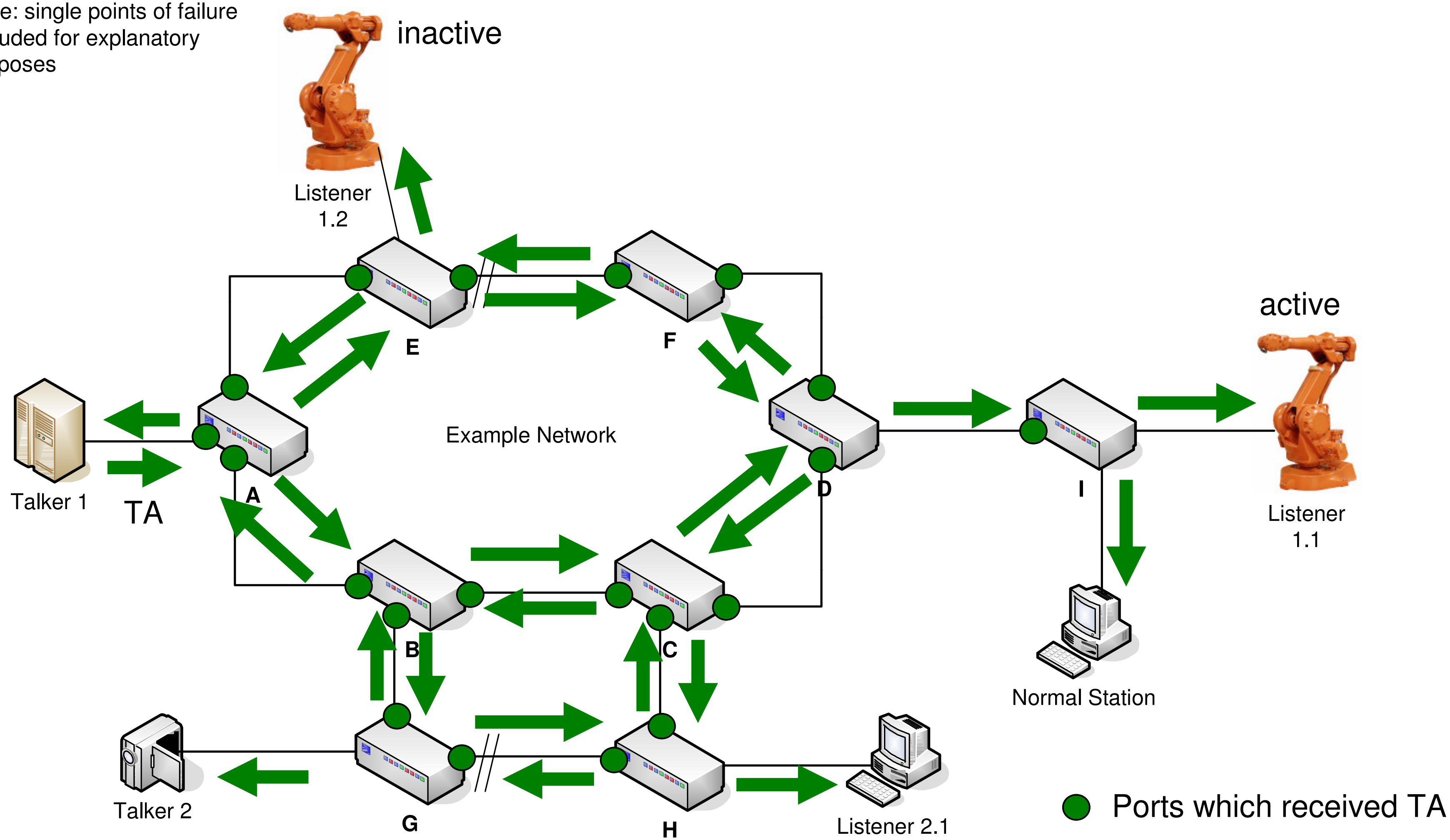
Note: single points of failure included for explanatory purposes



Talker advertisements are sent by bridges on all ports except the ports the same advertisement has been sent to already (and on which the same TA has been received)

Example network - part 1: talker advertise

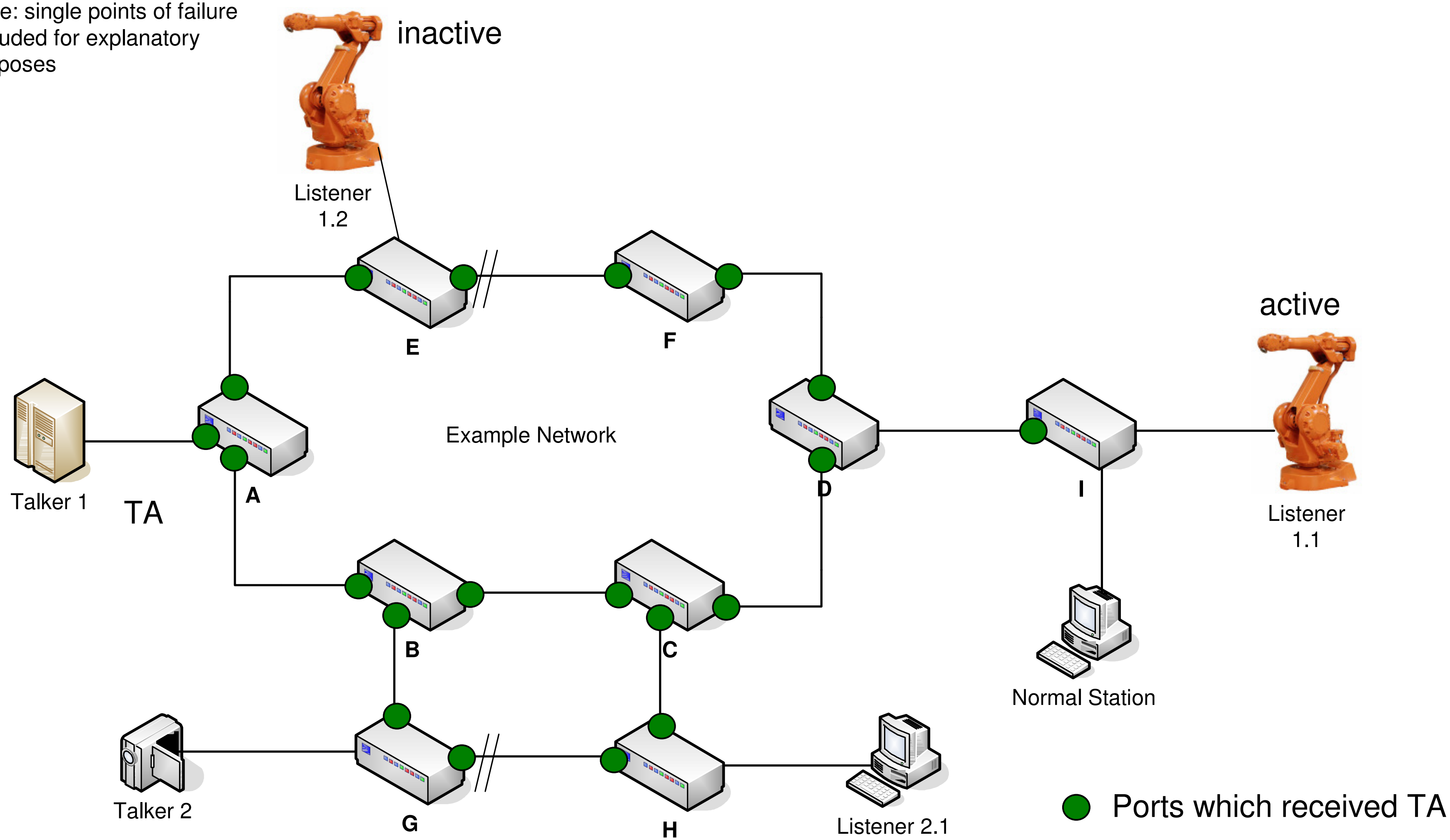
Note: single points of failure included for explanatory purposes



Talker advertisements are sent by bridges on all ports except the ports the same advertisement has been sent to already (and on which the same TA has been received)

Example network - part 1: talker advertise

Note: single points of failure included for explanatory purposes



Bridges now know on which ports they can „reach“ the talker (The ports on which they recieved the TA)

Part 2: listener ready

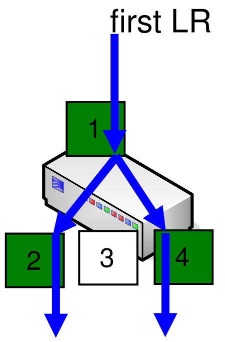
Bridge behaviour:

• A bridge that receives a listener ready and can identify the corresponding stream as redundant sends the listener ready to all ports it has not sent the listener ready before (except the receiving port) and on which it has received a corresponding Talker Advertise

• If a bridge has sent the listener ready to all ports, it drops all further listener ready frames for that particular stream ID (until after the next leave-all interval has passed)

•A bridge registers on which ports it received the listener ready

(essentially, it works the same way as the Talker Advertise, with exception of the ports that were not „marked“ by the TA in the previous step)



Note: Green ports have received a TA

Part 2: listener ready

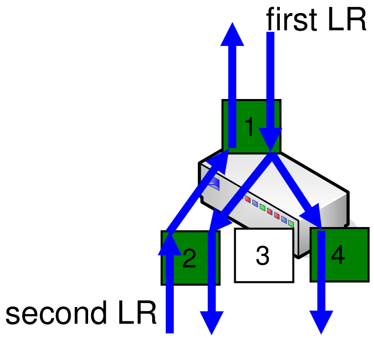
Bridge behaviour:

• A bridge that receives a listener ready and can identify the corresponding stream as redundant sends the listener ready to all ports it has not sent the listener ready before (except the receiving port) and on which it has received a corresponding Talker Advertise

• If a bridge has sent the listener ready to all ports, it drops all further listener ready frames for that particular stream ID (until after the next leave-all interval has passed)

•A bridge registers on which ports it received the listener ready

(essentially, it works the same way as the Talker Advertise, with exception of the ports that were not „marked" by the TA in the previous step)



Note: Green ports have received a TA

Part 2: listener ready

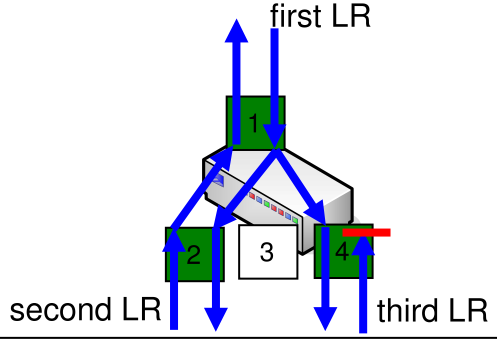
Bridge behaviour:

• A bridge that receives a listener ready and can identify the corresponding stream as redundant sends the listener ready to all ports it has not sent the listener ready before (except the receiving port) and on which it has received a corresponding Talker Advertise

• If a bridge has sent the listener ready to all ports, it drops all further listener ready frames for that particular stream ID (until after the next leave-all interval has passed)

•A bridge registers on which ports it received the listener ready

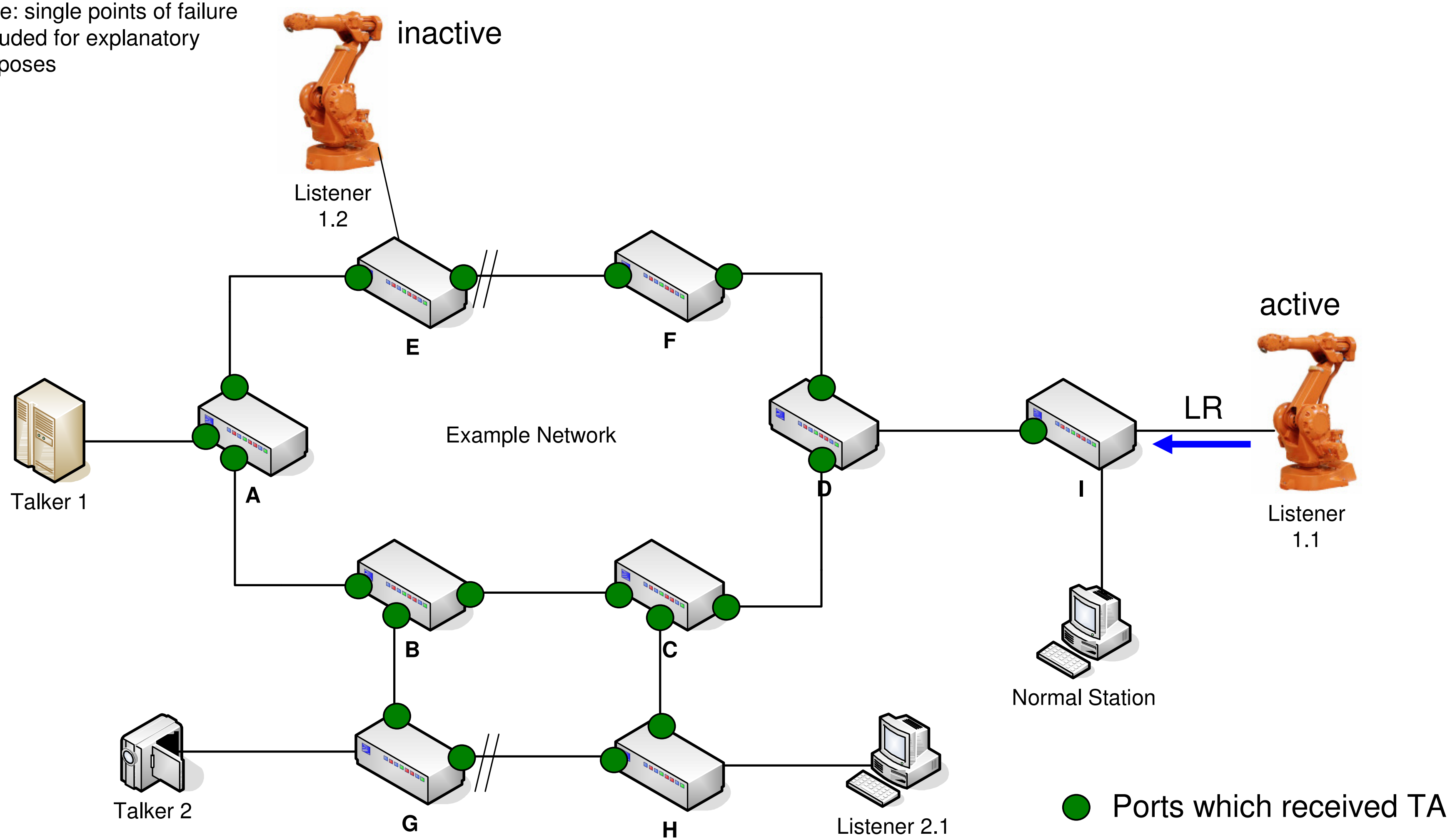
(essentially, it works the same way as the Talker Advertise, with exception of the ports that were not „marked“ by the TA in the previous step)



Note: Green ports have received a TA

Part 2: listener ready

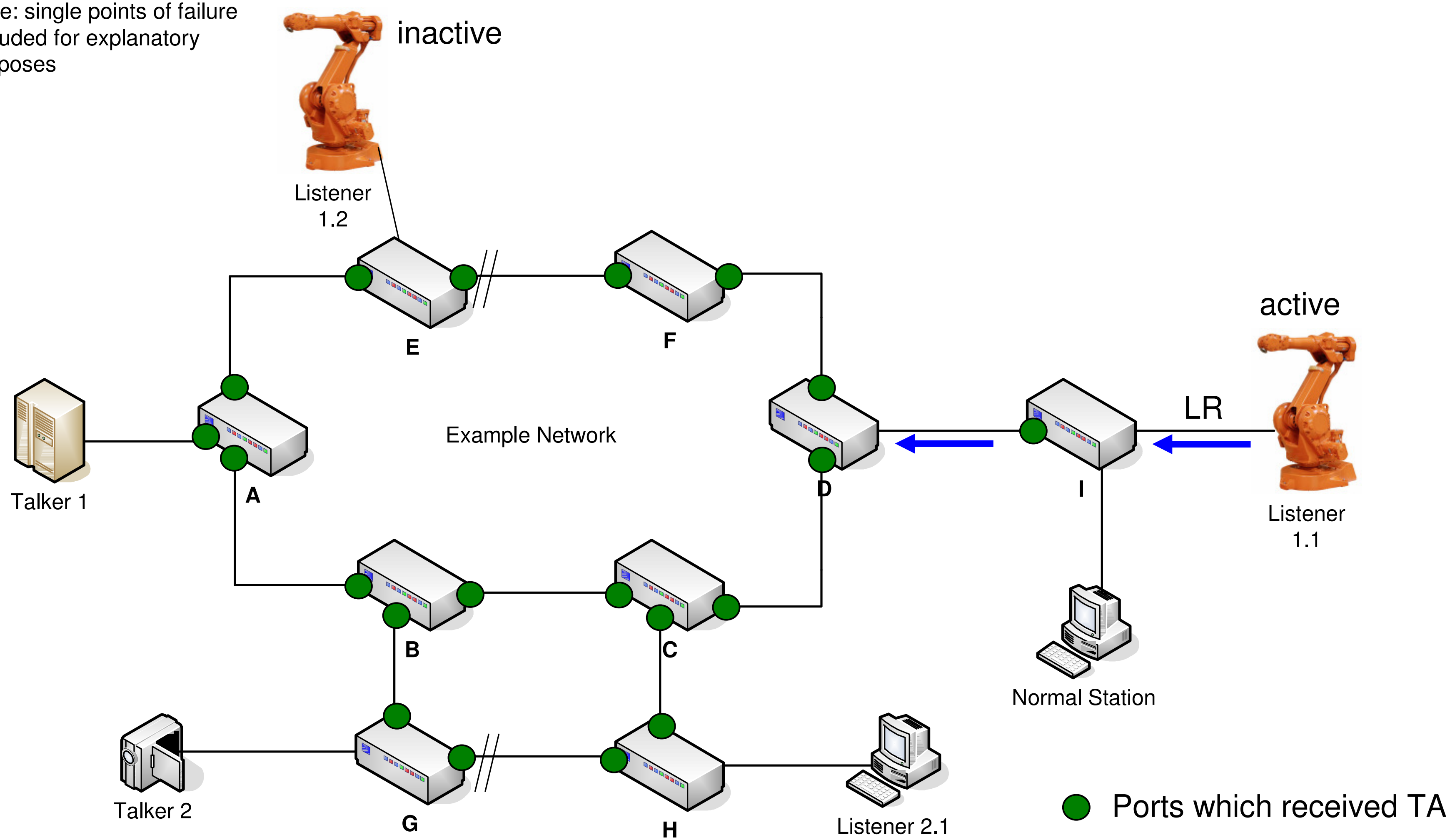
Note: single points of failure included for explanatory purposes



Bridges send „listener ready“ on all ports they received a „talker advertise“ (except the receiving port)

Part 2: listener ready

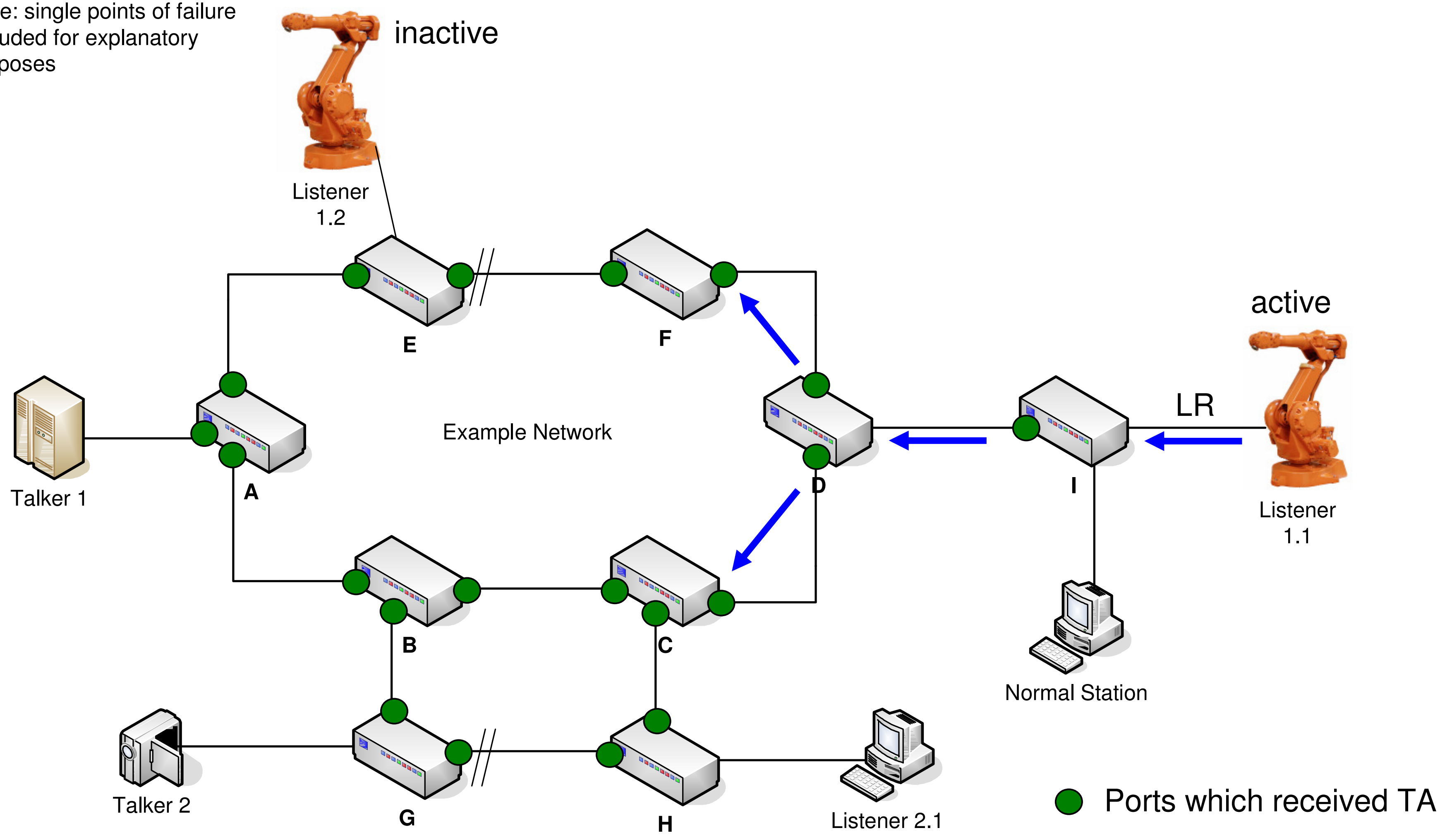
Note: single points of failure included for explanatory purposes



Bridges send „listener ready“ on all ports they received a „talker advertise“ (except the receiving port)

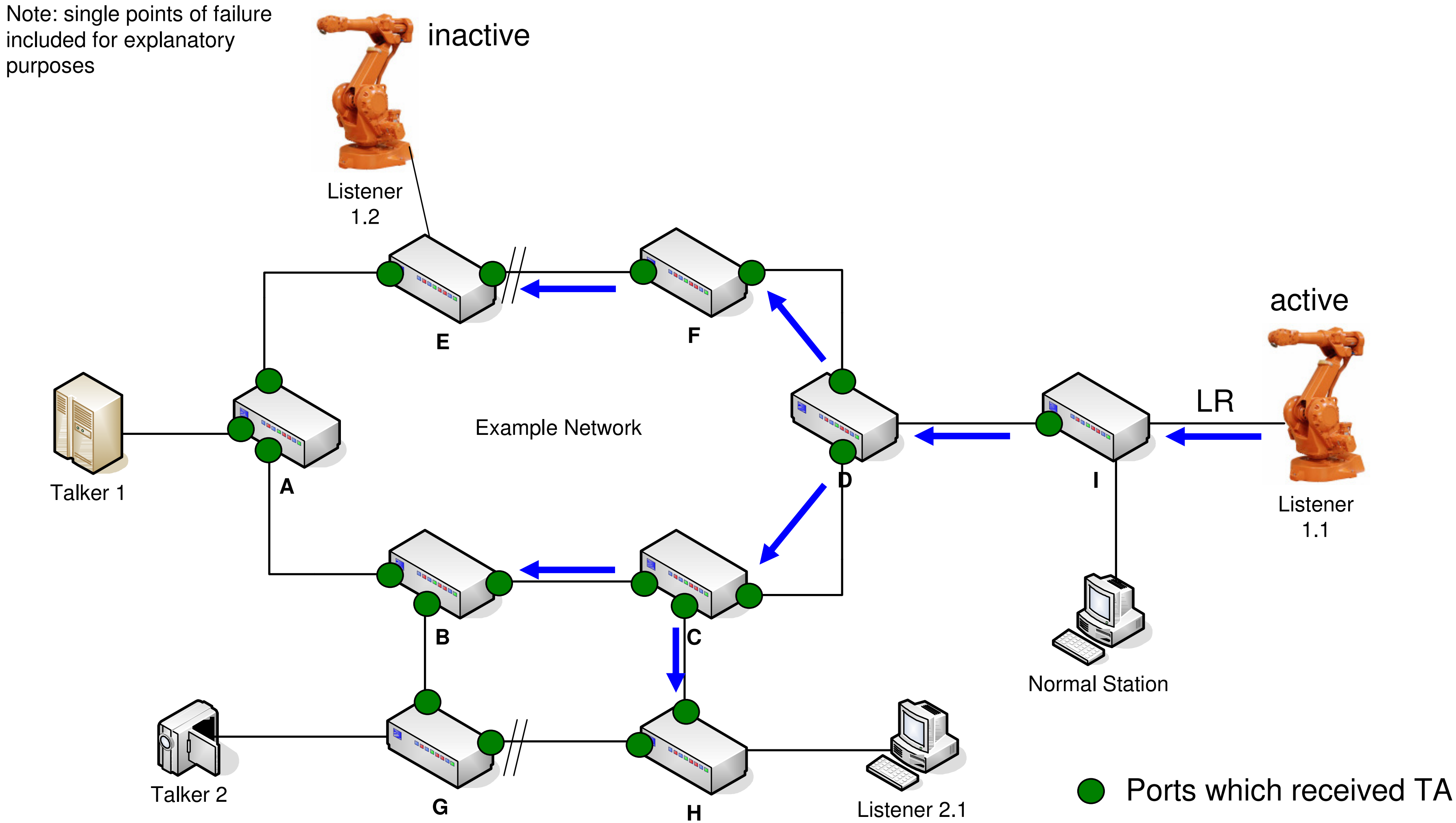
Part 2: listener ready

Note: single points of failure included for explanatory purposes



Bridges send „listener ready" on all ports they received a „talker advertise" (except the receiving port)

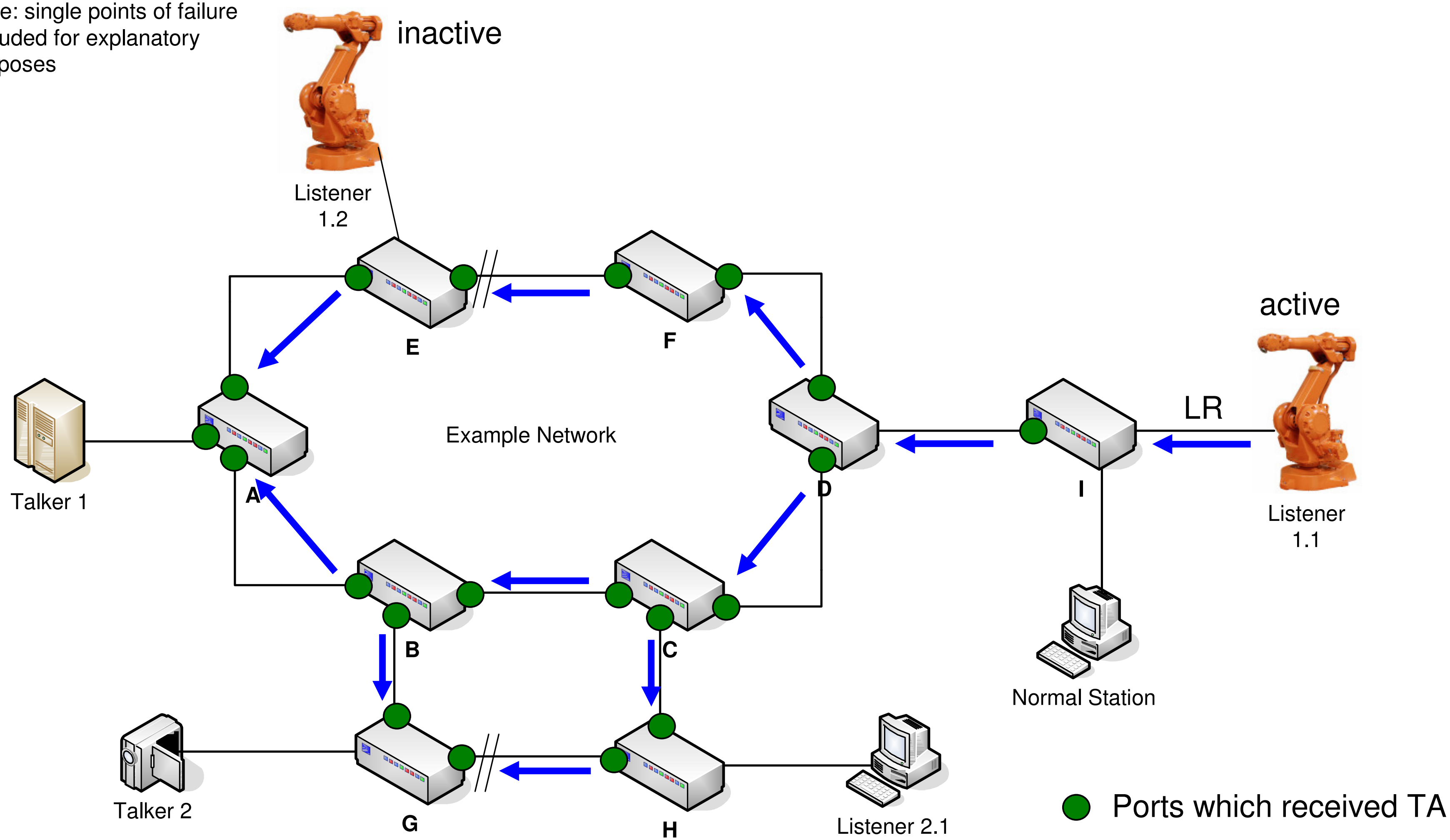
Part 2: listener ready



Bridges send „listener ready“ on all ports they received a „talker advertise“ (except the receiving port)

Part 2: listener ready

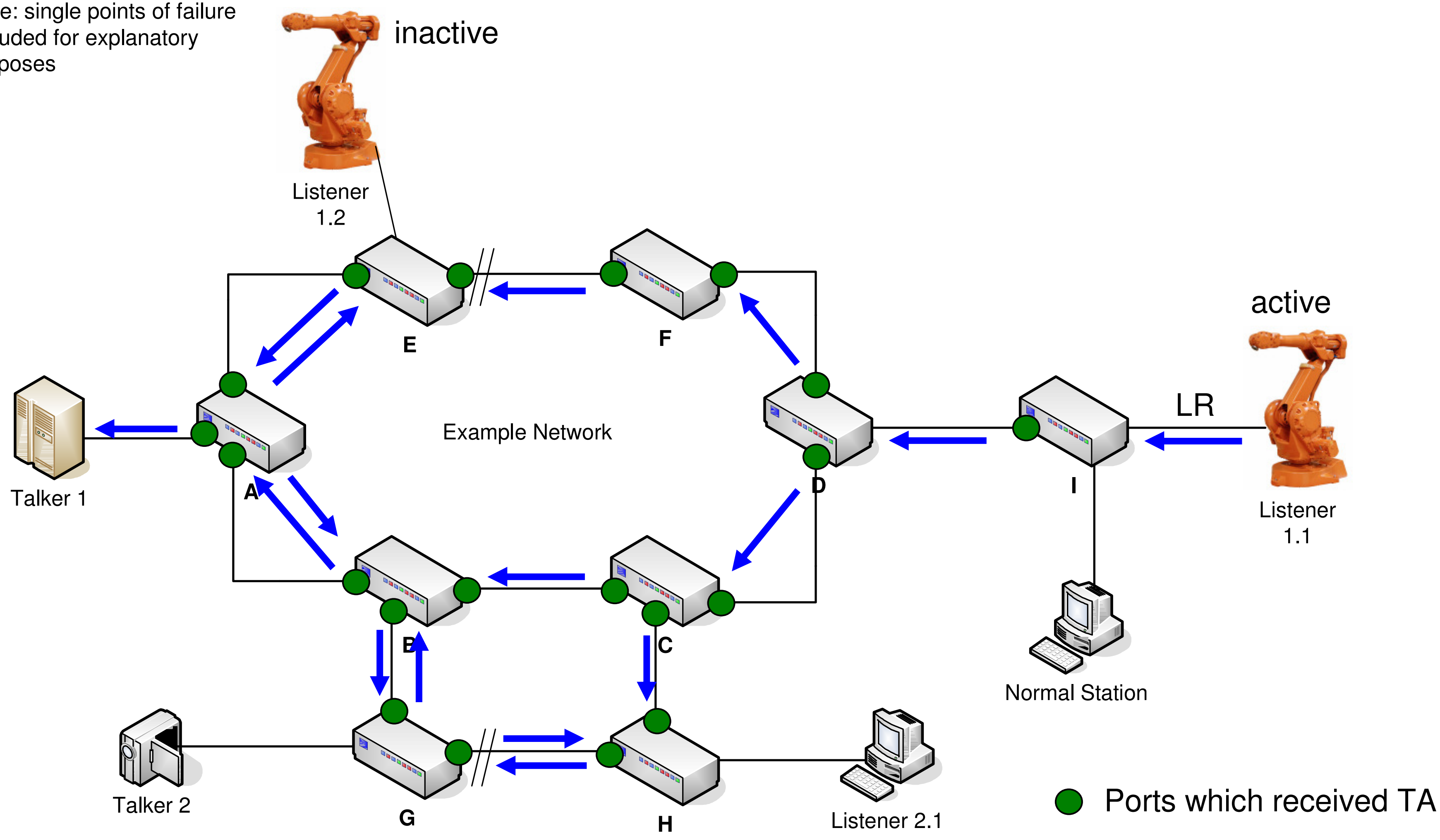
Note: single points of failure included for explanatory purposes



Bridges send „listener ready“ on all ports they received a „talker advertise“ (except the receiving port)

Part 2: listener ready

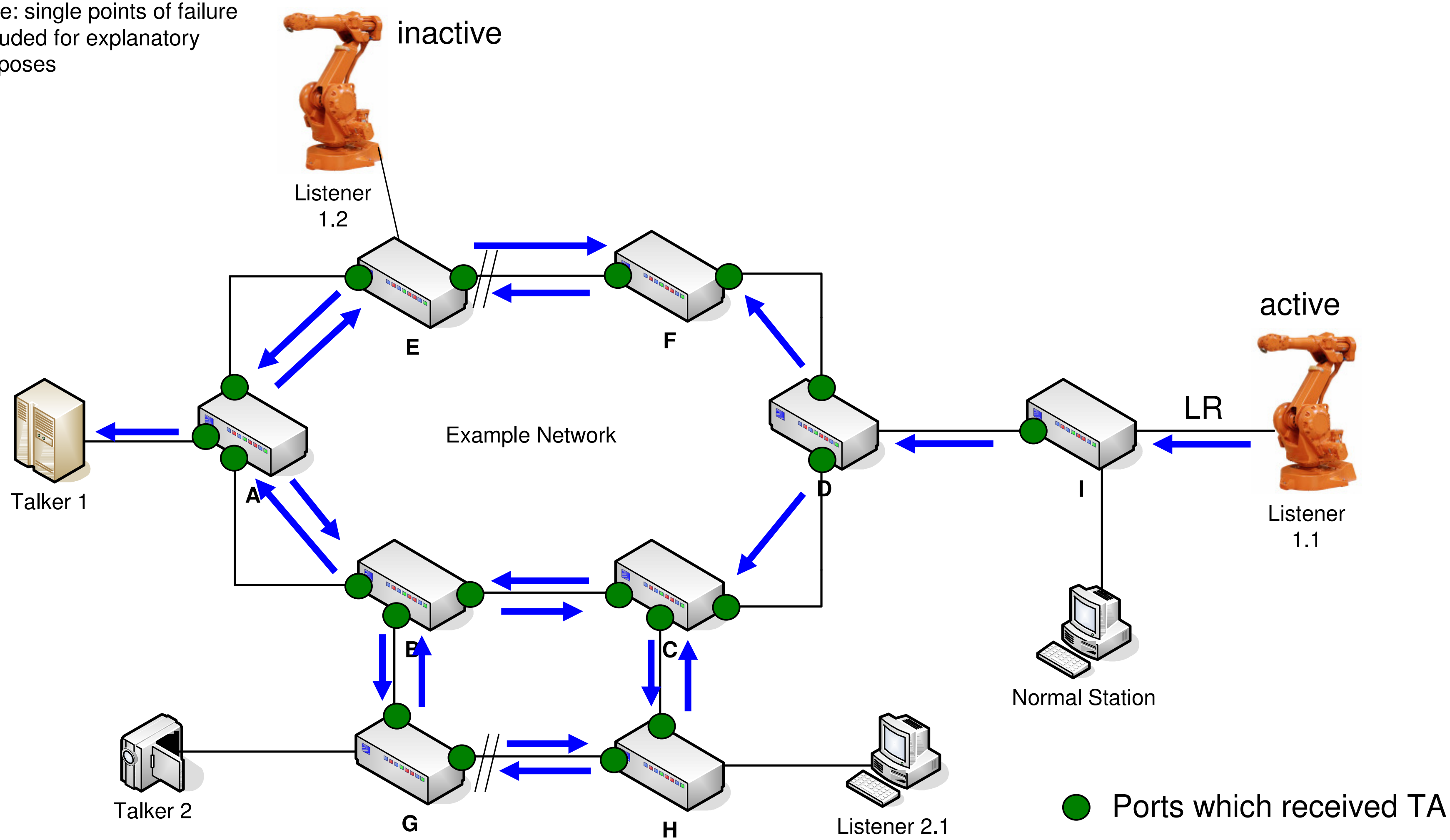
Note: single points of failure included for explanatory purposes



Bridges send „listener ready“ on all ports they received a „talker advertise“ (except the receiving port)

Part 2: listener ready

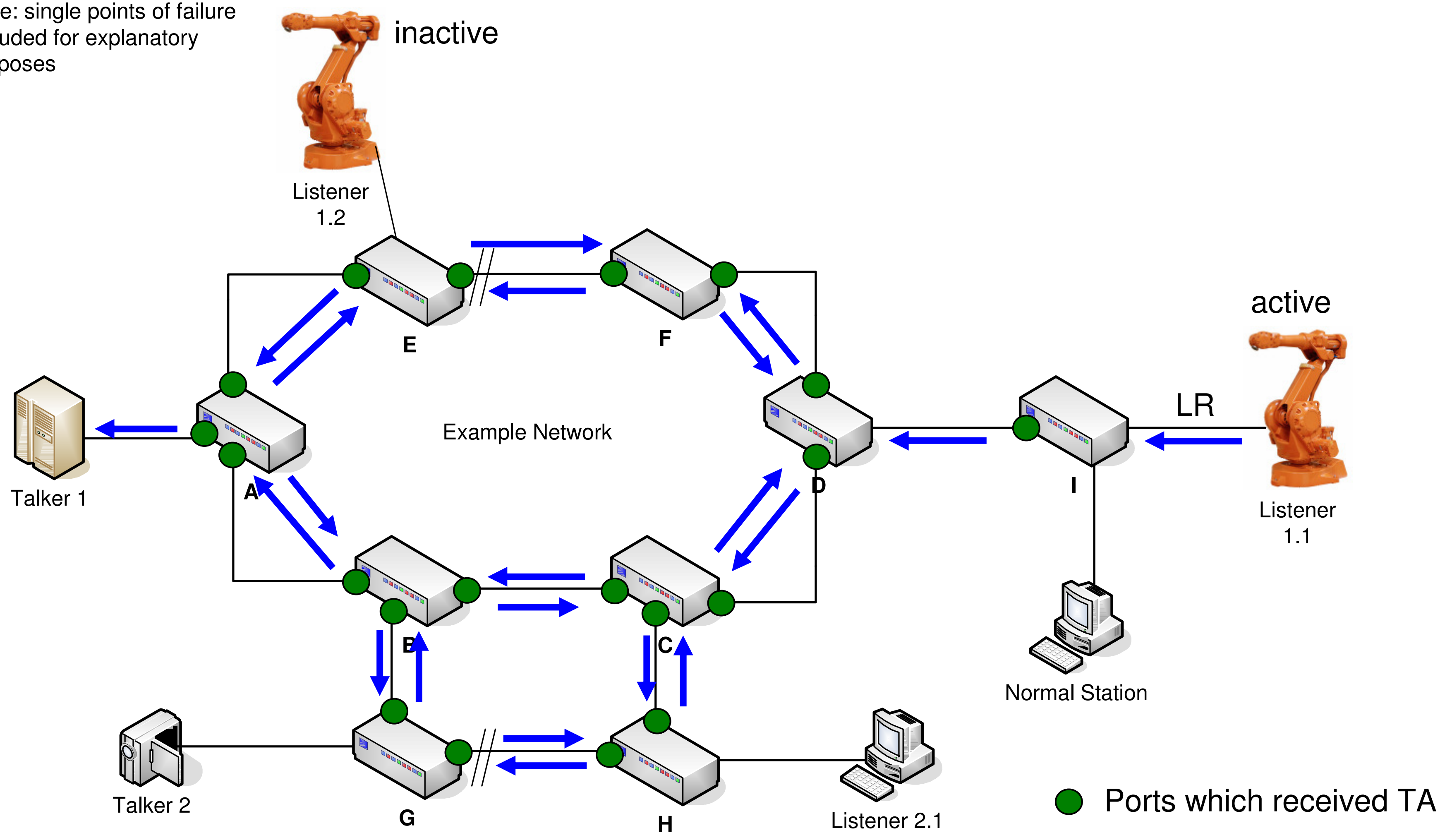
Note: single points of failure included for explanatory purposes



Bridges send „listener ready" on all ports they received a „talker advertise" (except the receiving port)

Part 2: listener ready

Note: single points of failure included for explanatory purposes



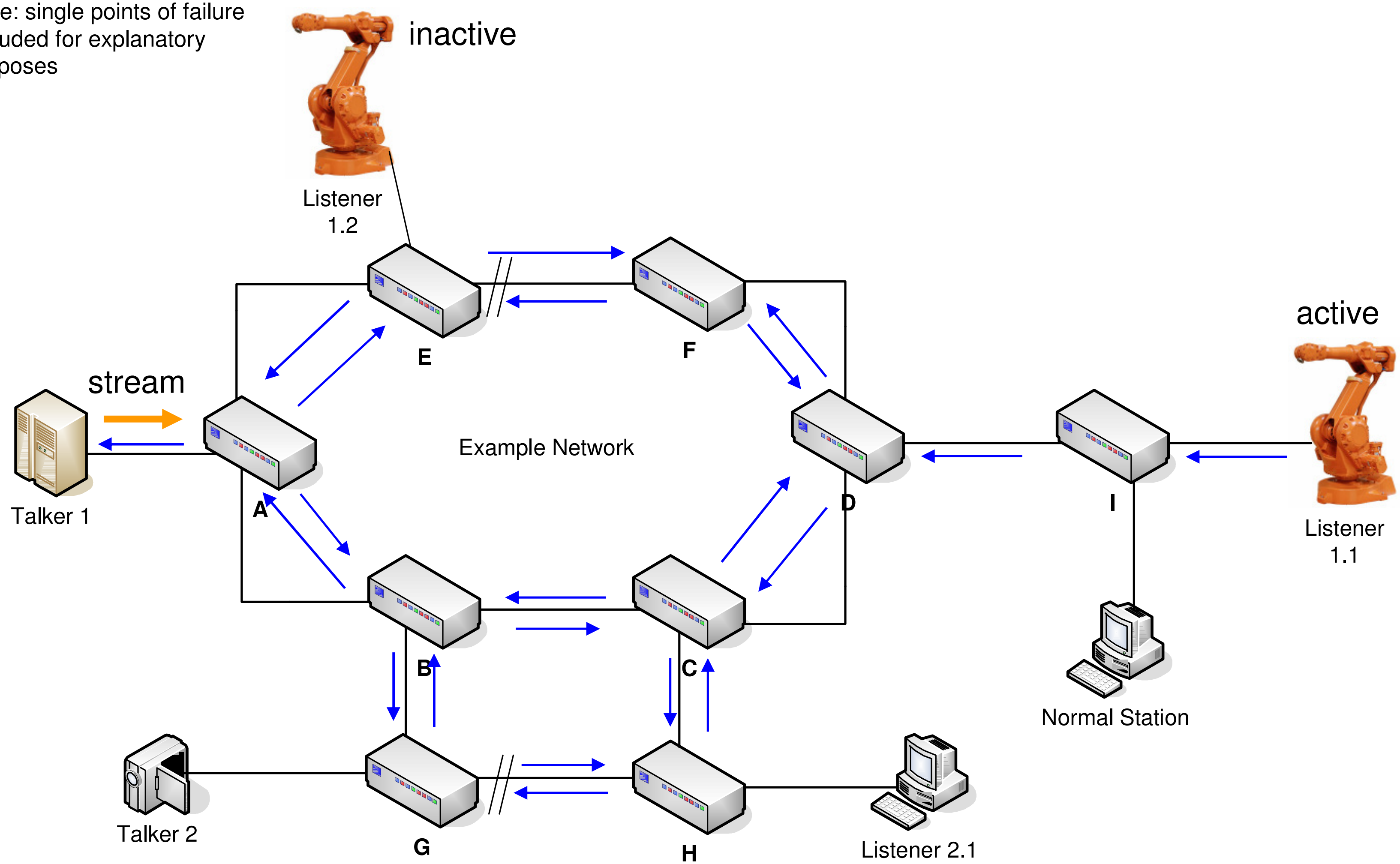
Bridges send „listener ready“ on all ports they received a „talker advertise“ (except the receiving port)

Part 3: stream transmission

Stream transmission

Part 3: stream transmission

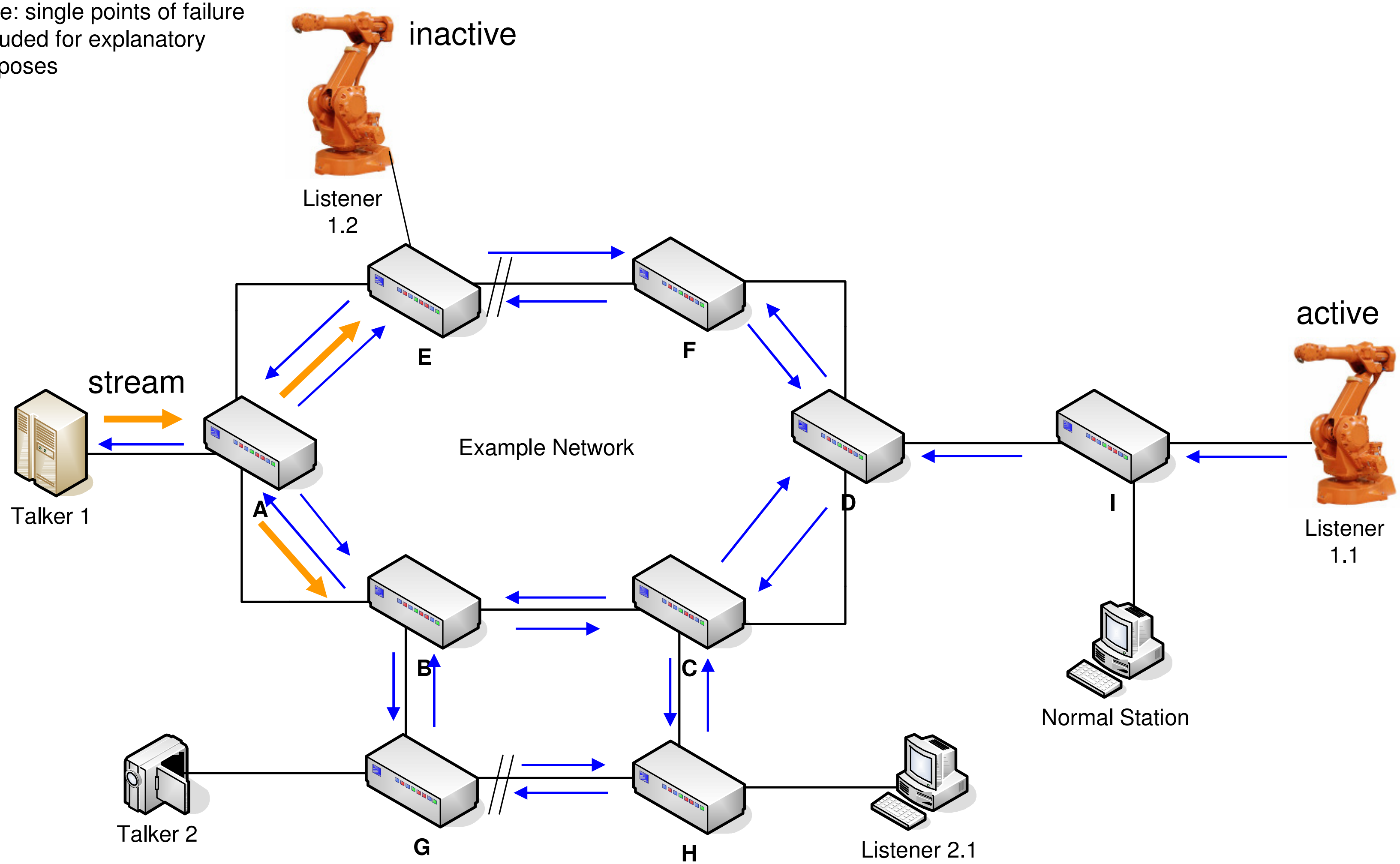
Note: single points of failure included for explanatory purposes



The stream frames are forwarded in the opposite direction of the received listener ready frames

Part 3: stream transmission

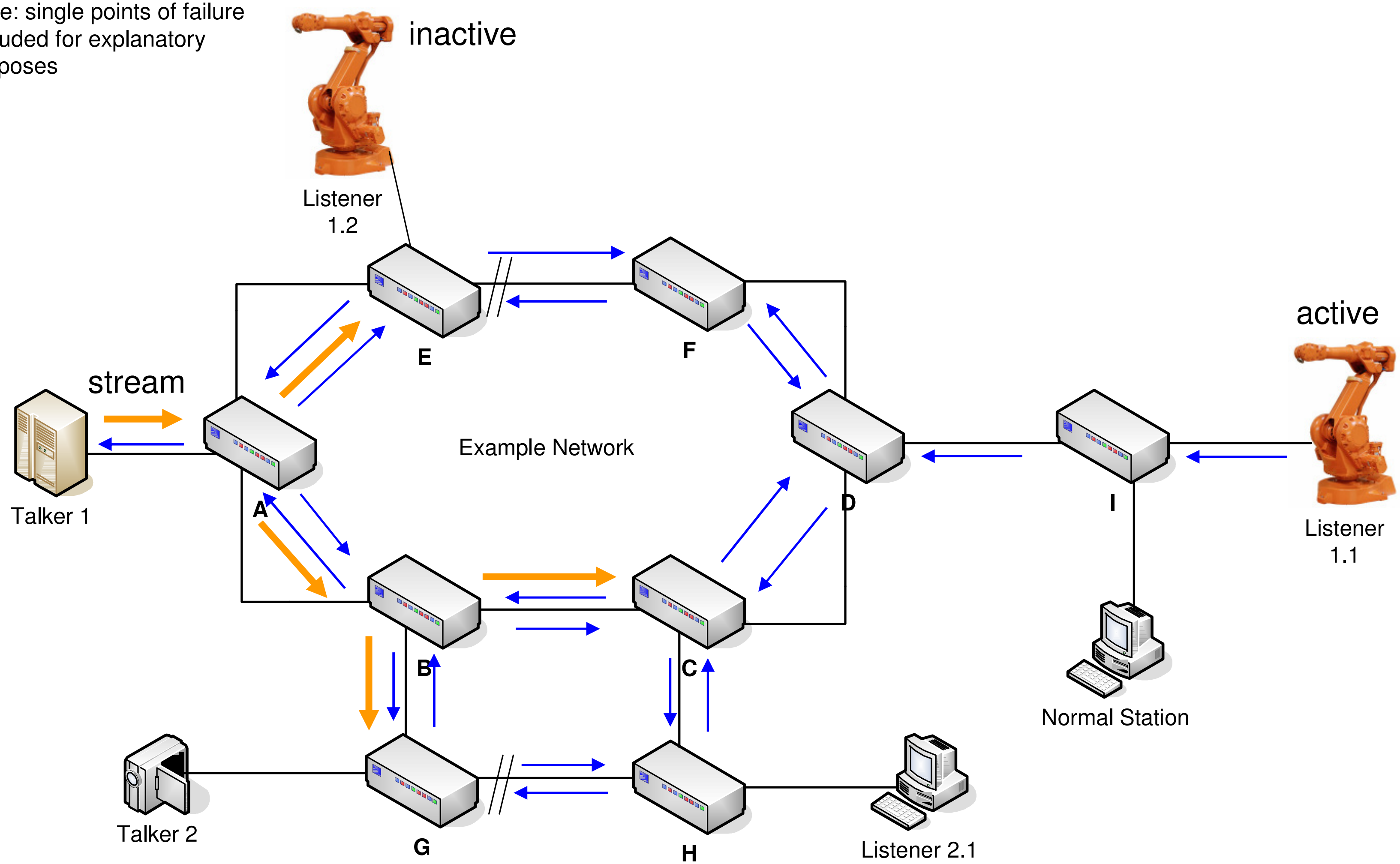
Note: single points of failure included for explanatory purposes



The stream frames are forwarded in the opposite direction of the received listener ready frames

Part 3: stream transmission

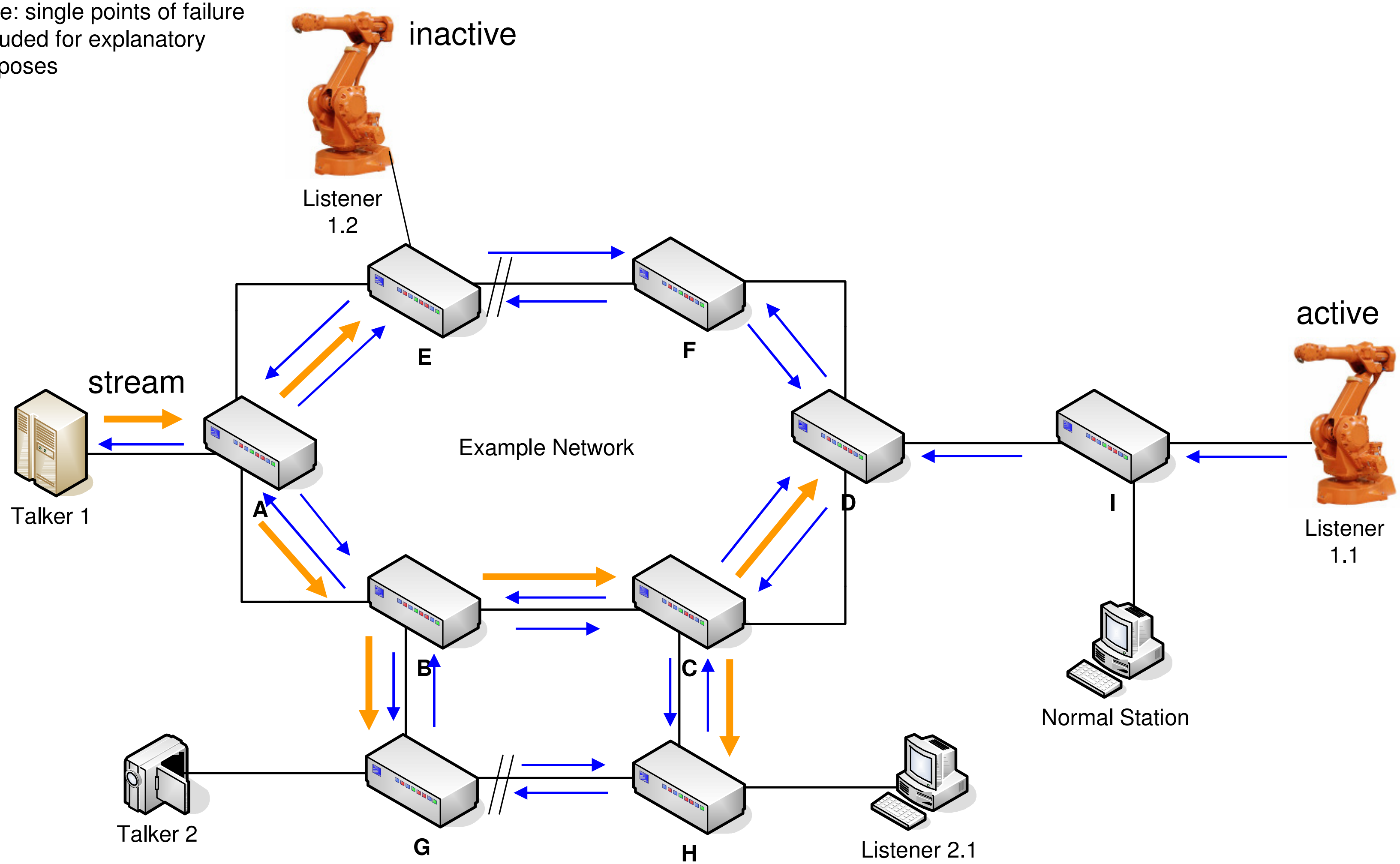
Note: single points of failure included for explanatory purposes



The stream frames are forwarded in the opposite direction of the received listener ready frames

Part 3: stream transmission

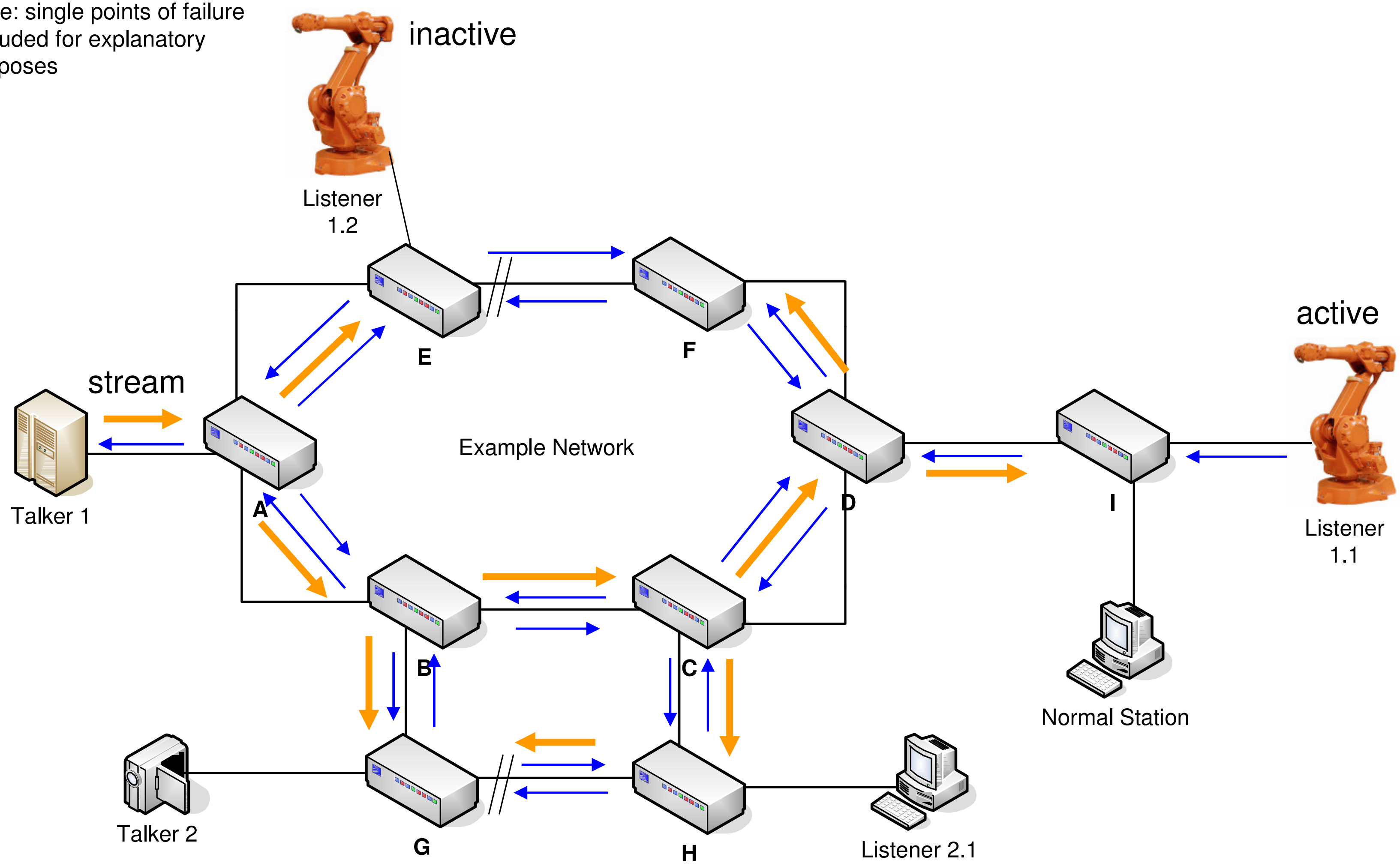
Note: single points of failure included for explanatory purposes



The stream frames are forwarded in the opposite direction of the received listener ready frames

Part 3: stream transmission

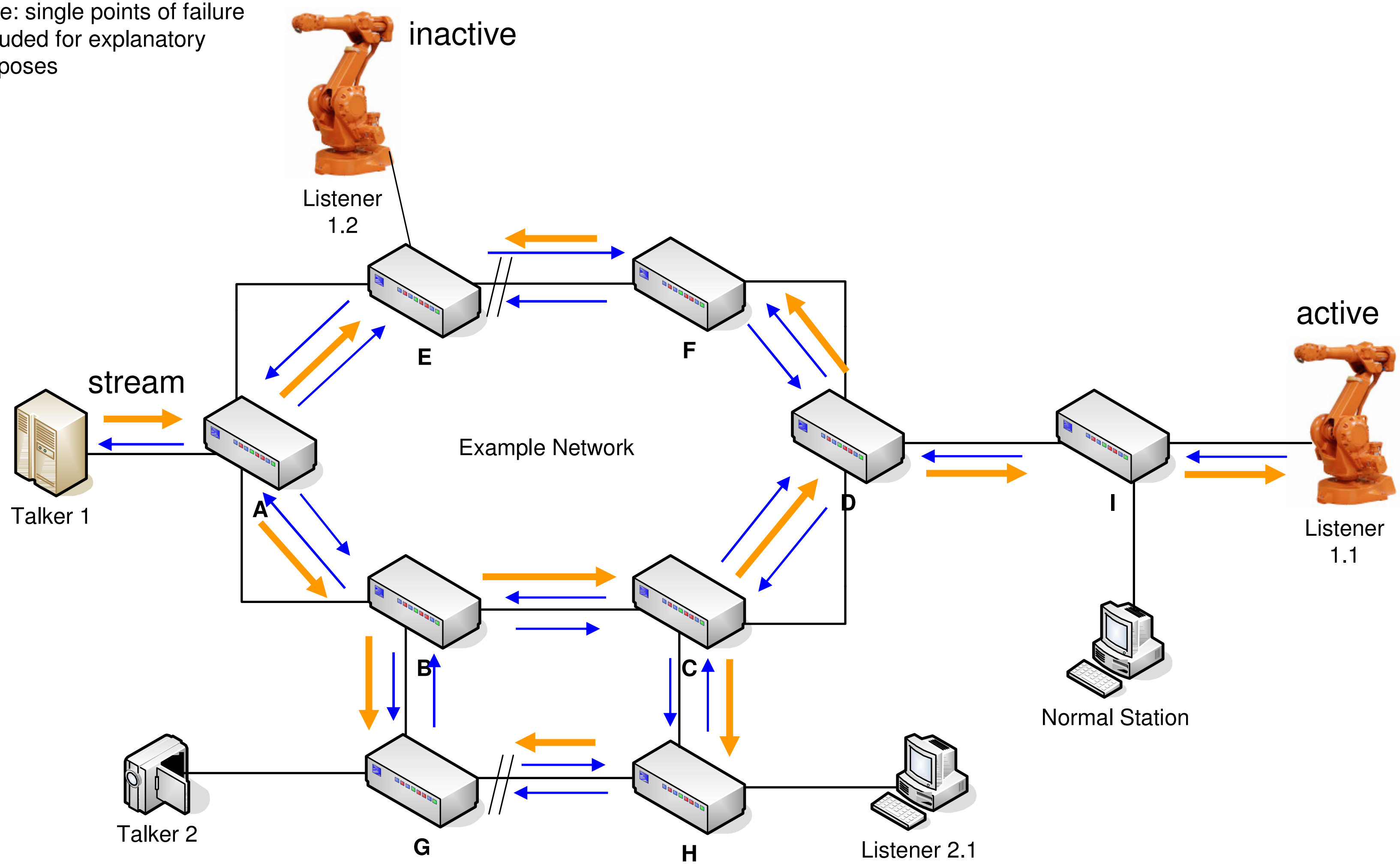
Note: single points of failure included for explanatory purposes



The stream frames are forwarded in the opposite direction of the received listener ready frames

Part 3: stream transmission

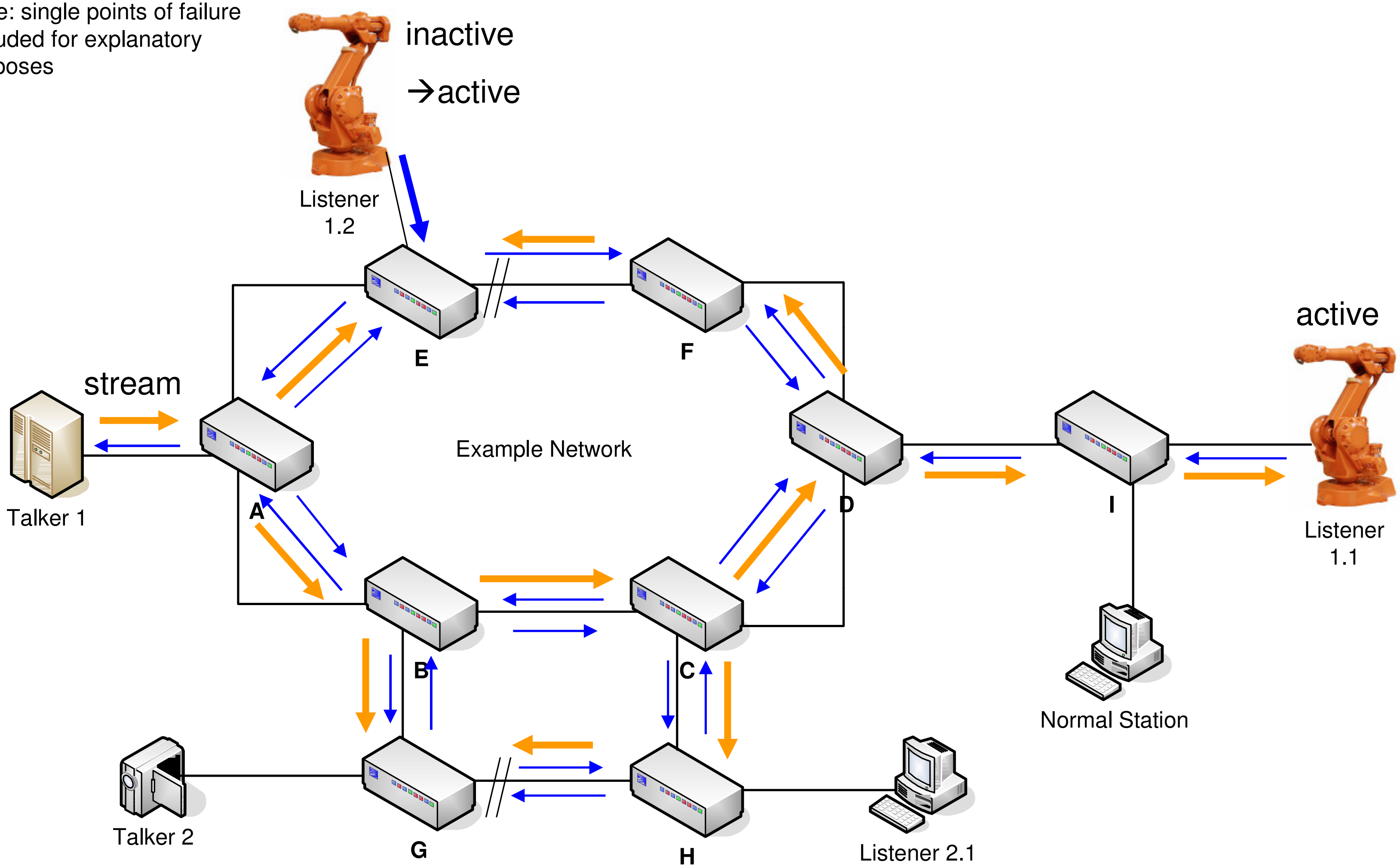
Note: single points of failure included for explanatory purposes



The stream frames are forwarded in the opposite direction of the received listener ready frames

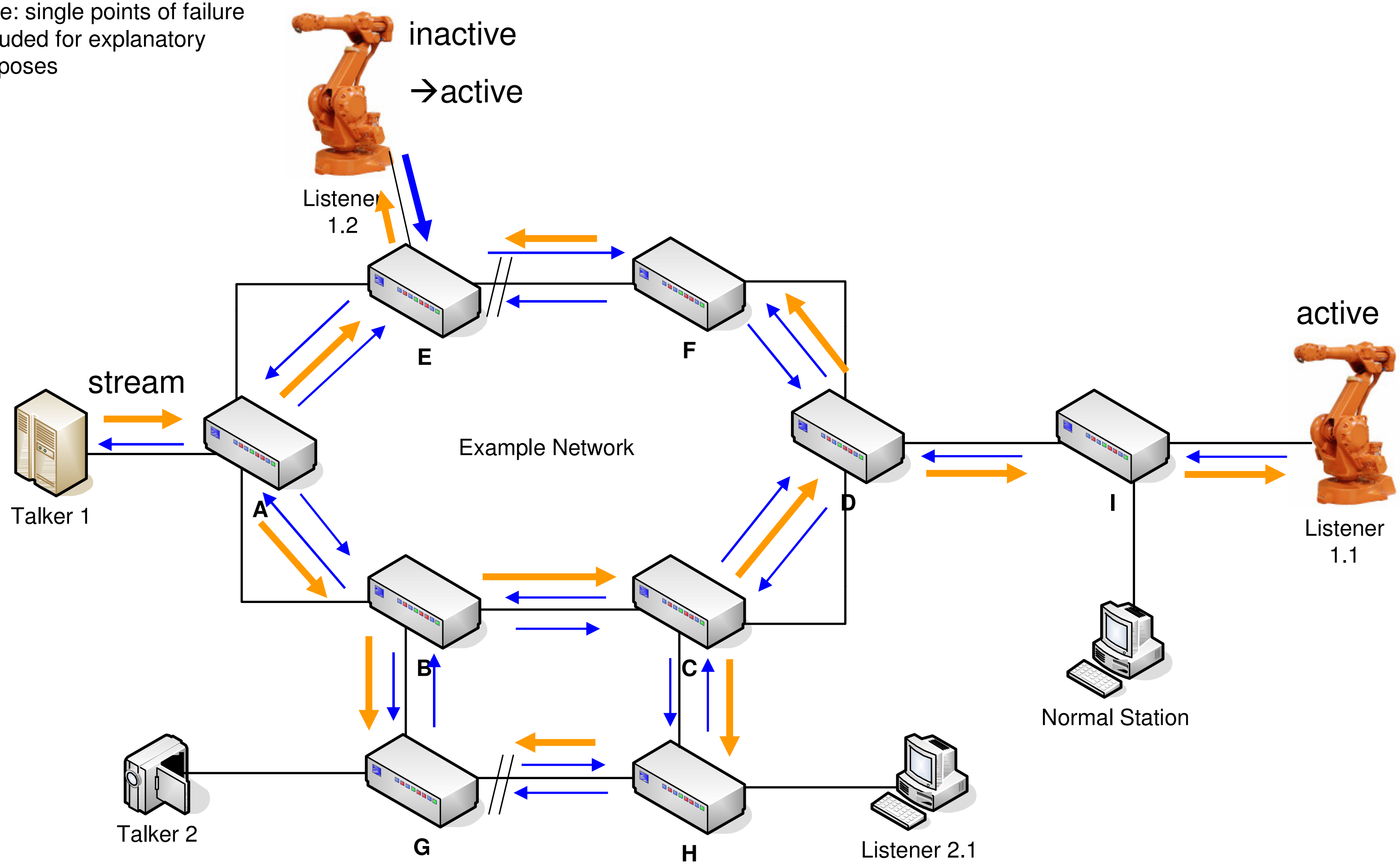
Part 3: another listener joins

Note: single points of failure included for explanatory purposes



Part 3: another listener joins

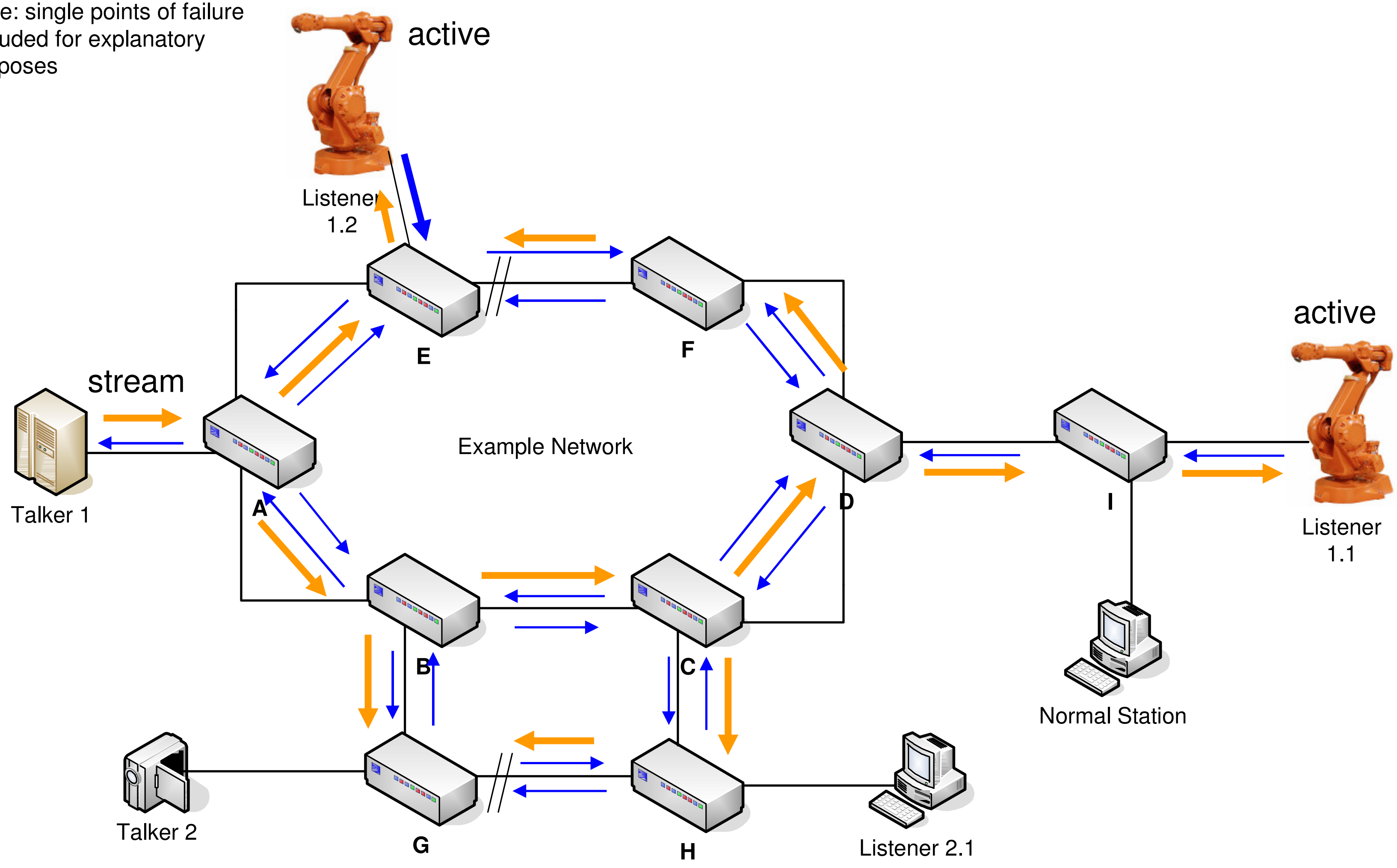
Note: single points of failure included for explanatory purposes



Listener 1.2 is „supplied“ redundantly automatically

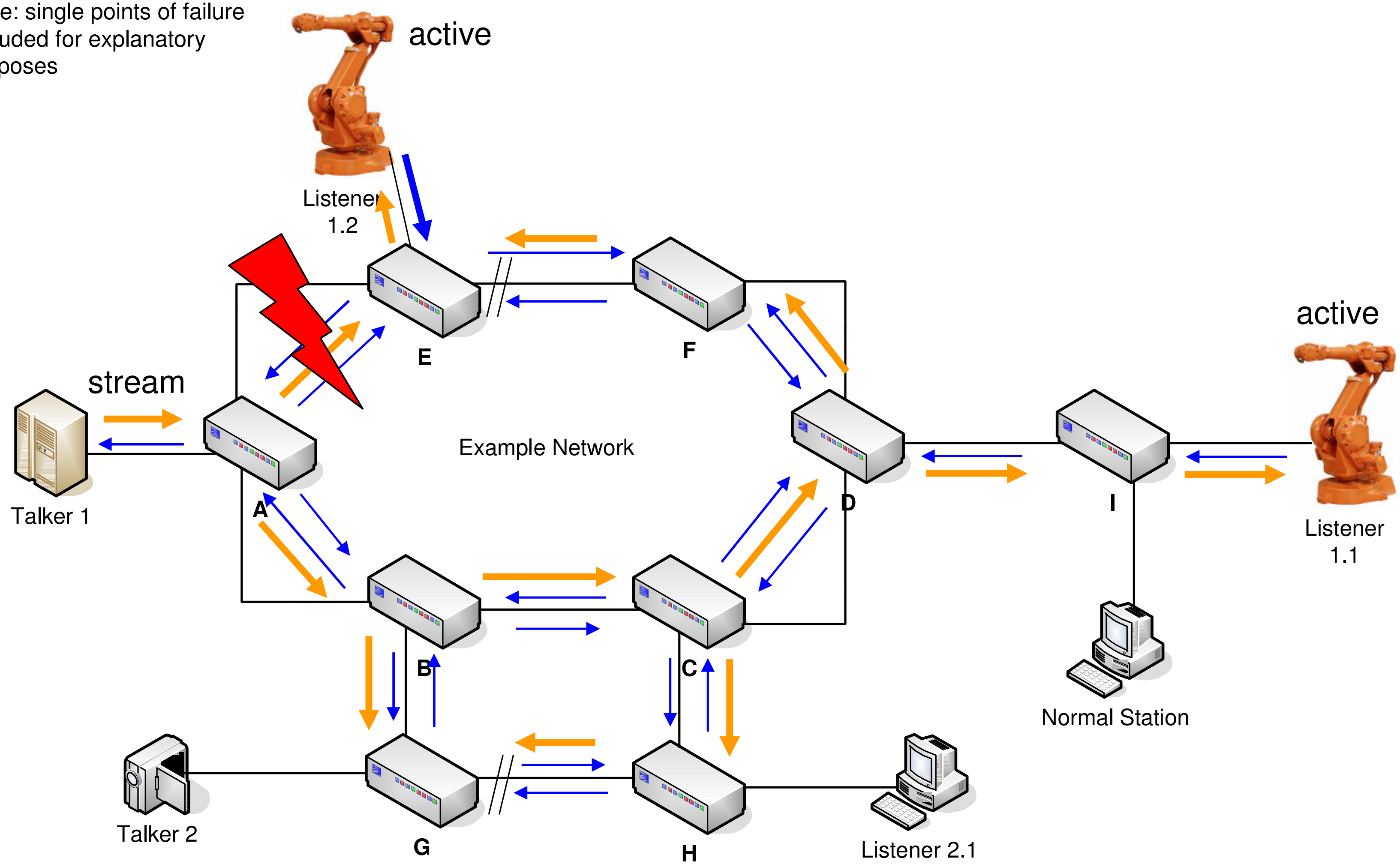
What happens in case of a fault?

Note: single points of failure included for explanatory purposes

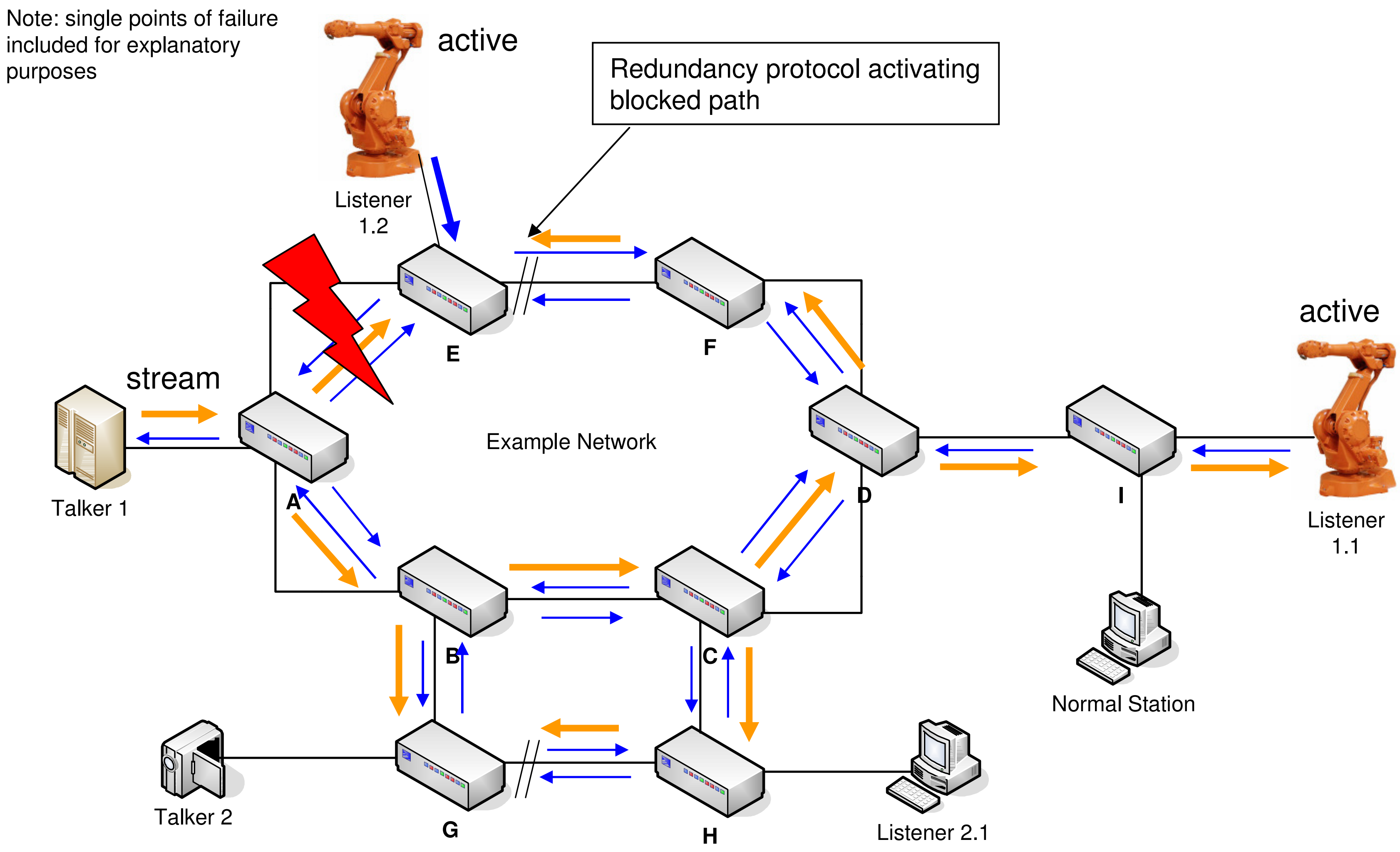


What happens in case of a fault?

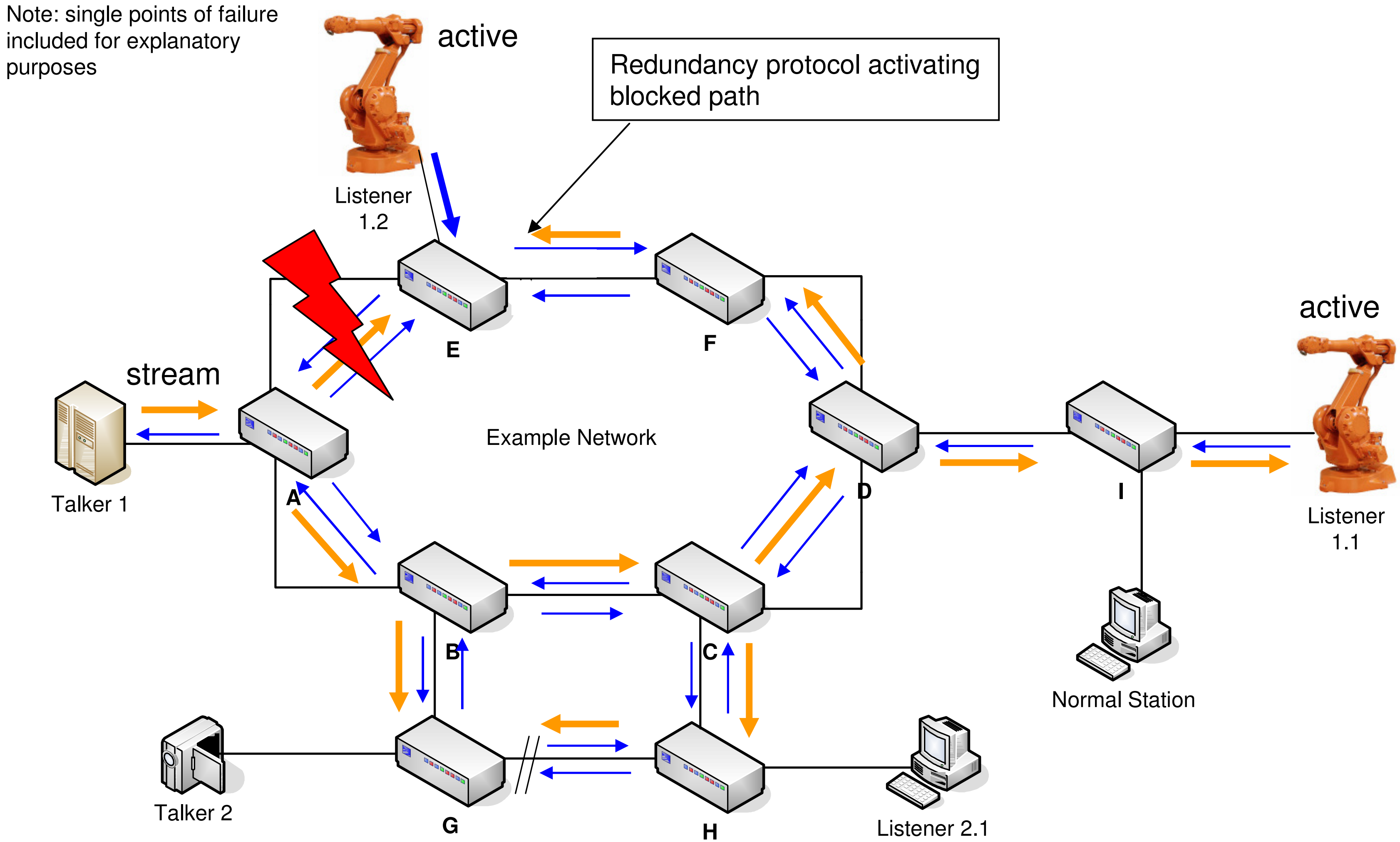
Note: single points of failure included for explanatory purposes



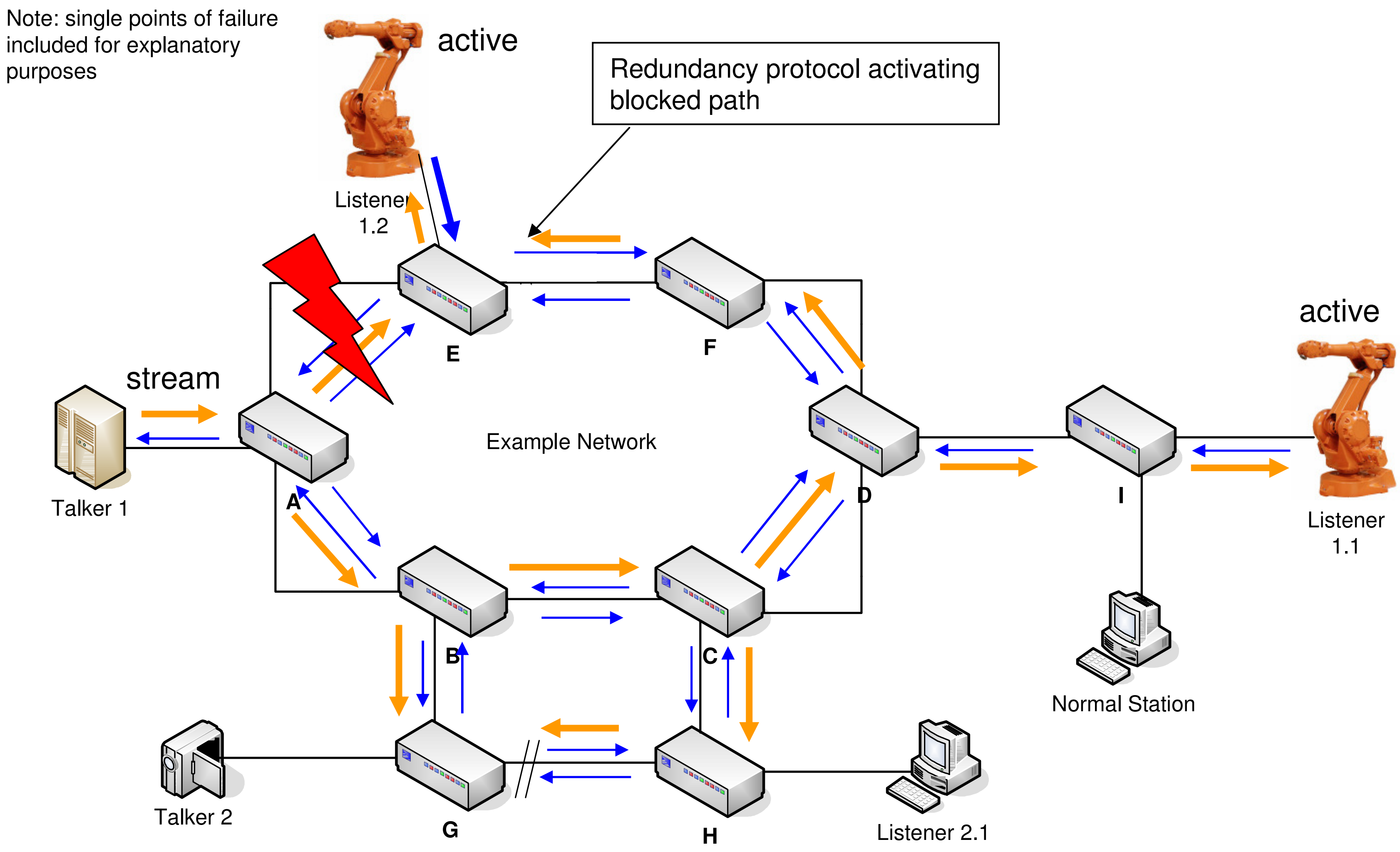
What happens in case of a fault?



What happens in case of a fault?



What happens in case of a fault?



Reconfiguration time will be equal or very close to redundancy control protocol reconfiguration time

Further insight into possibilities of realization -

Differences between networks with and without interruption

Technologies with network interruption

	redundant links	redundant networks
with network interruption	✓	✓
without network interruption		

The previously described network was a network with interruption (hence the discarding ports in the example network):

Discarding ports are ignored for stream registrations, but are observed for stream frame transmission

Reservations happen on all paths, but stream frames will not necessarily travel the whole path, but will terminate at discarding ports

In case of a fault and a recovery through a redundancy protocol, after the discarding port is put into forwarding state, the stream transmission will immediately resume

→ This reduces the total network reconfiguration time to a value that is equal (or only marginally higher) thant the reconfiguration time of the redundancy protocol

Technologies without network interruption

	redundant links	redundant networks
with network interruption		
without network interruption	✓	✓

Mechanisms like HSR and PRP generate and consume redundant frames by themselves.

→ Method described above also applicaple to protocols like HSR/PRP, with the change that in case of a fault, there is no reconfiguration time at all, because the redundancy control protocol realizes a seamless switchover from one path to the other path

Stream path restrictions

Further insight into possibilities of realization -

Restricting reservations to certain paths

Stream path restrictions

For certain application fields, arbitrary paths may not be the best choice or may not be feasible.

To accomodate this requirement, additional management interfaces should be made available to enable or disable redundancy operation.

This could e.g. be defined as a MIB parameter

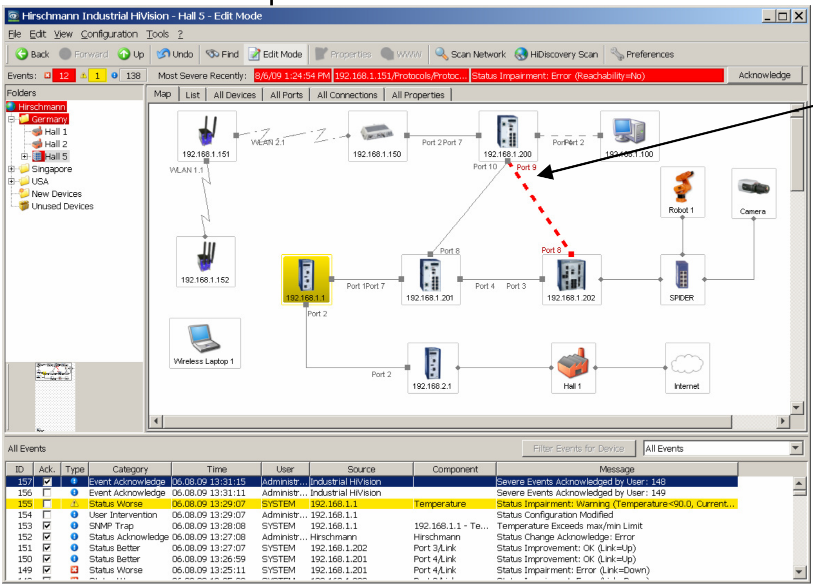
For ports that have redundancy enabled, the bridge behaves as described above, disabled ports do not participate in redundant stream transmission. (essentially behave as if the link were down for redundant stream handling)

Stream path restrictions

For an RSTP network, e.g. all bridge ports that have RSTP enabled will propably also have redundant operation enabled.

Other redundancy control protocols, e.g. those used in industrial communication systems, can also map their redundant links/ports directly, e.g. through HMI/SCADA

HMI/SCADA example

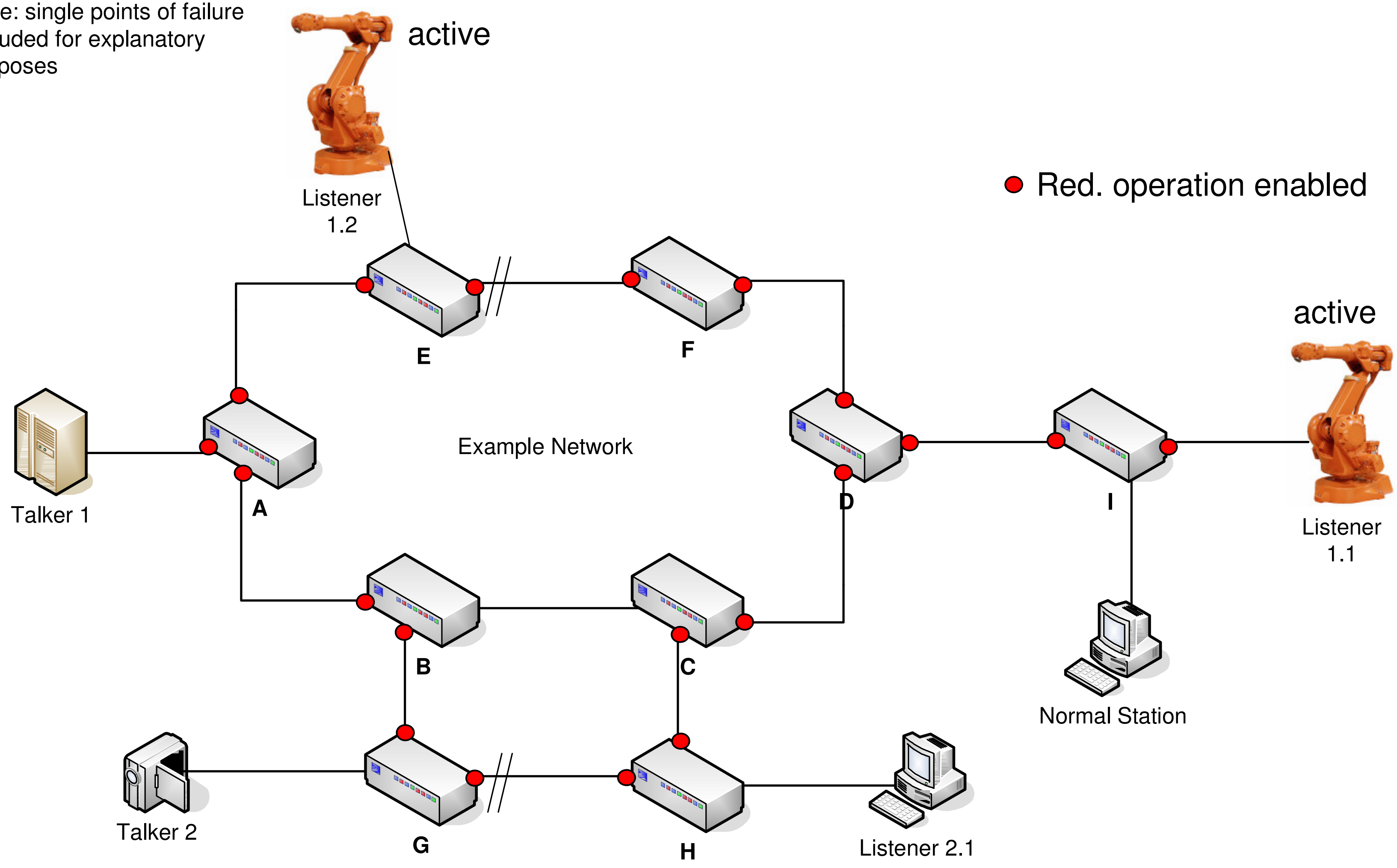


Semi – automatical or automatical redundancy protocol configuration (e.g. IEC 62439-2 MRP ring ports)

Redundancy control protocol port information maps directly to MSRP redundancy port operation

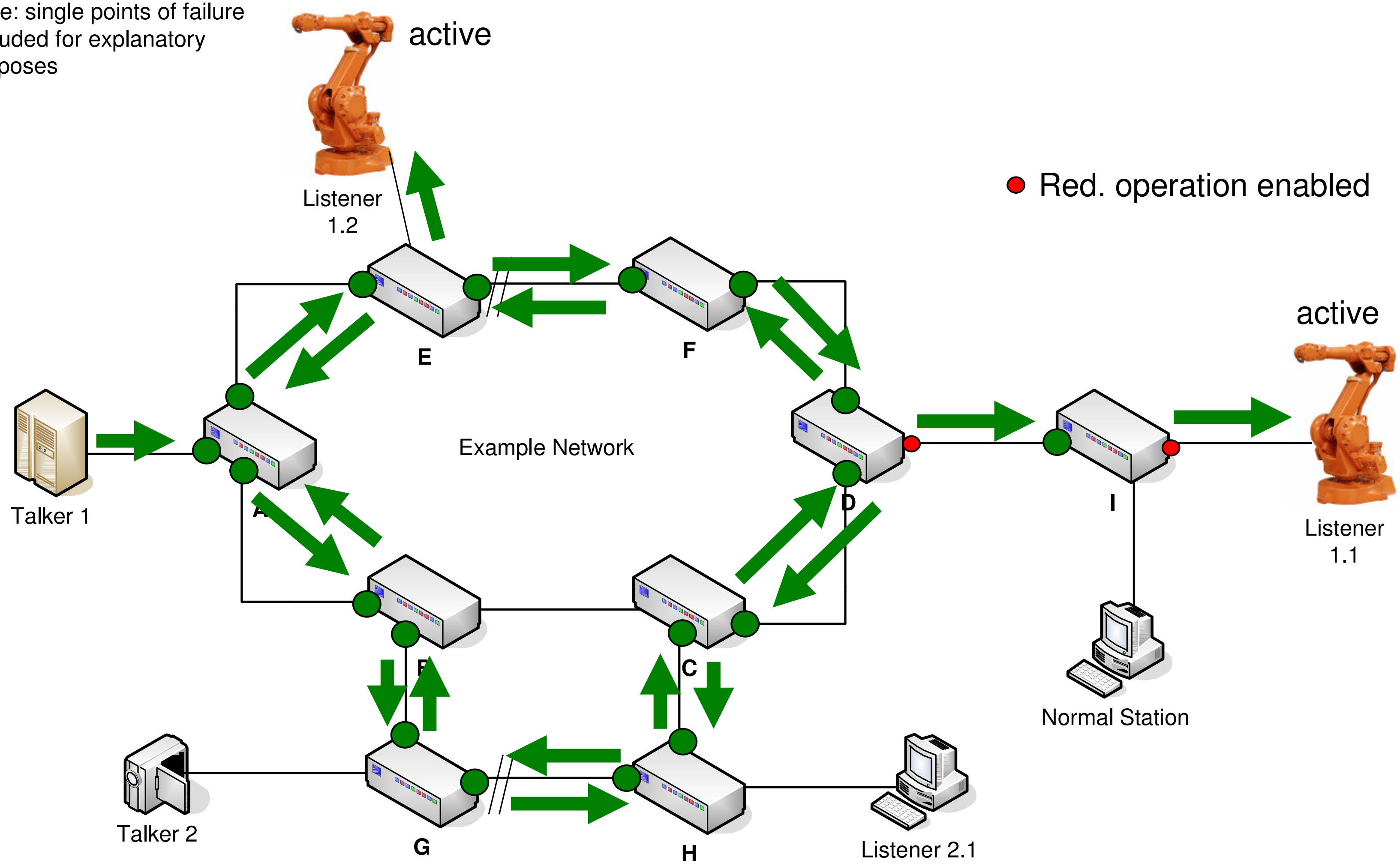
Stream path restrictions

Note: single points of failure included for explanatory purposes



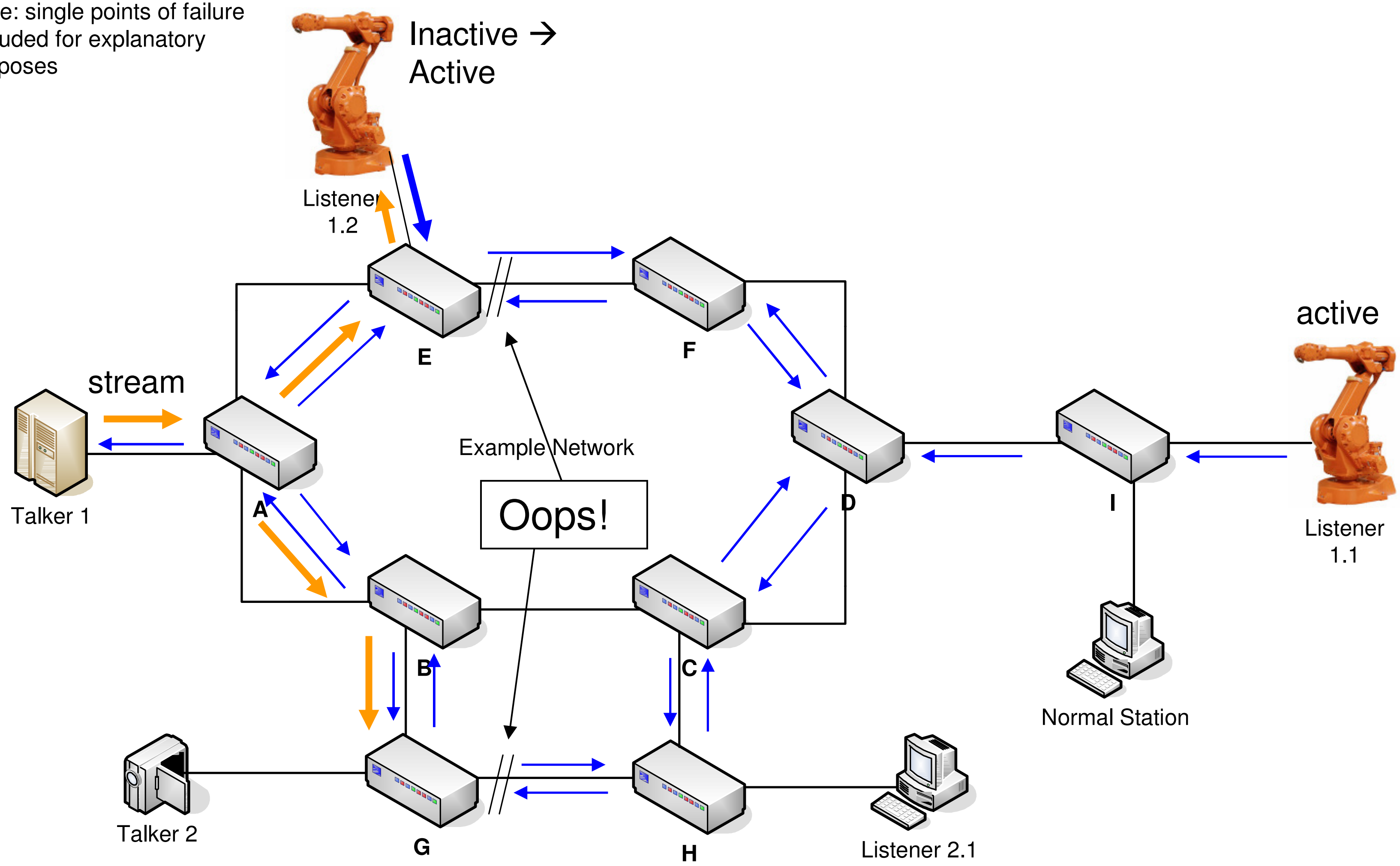
Stream path restrictions

Note: single points of failure included for explanatory purposes



Stream path restrictions

Note: single points of failure included for explanatory purposes



With configuration, the active topology devised by the redundancy protocol needs to be observed! → Must be done by application

FIN

Thank you for your attention!

Backup slides

Configuration of redundant paths

Other possibilities for path selection:

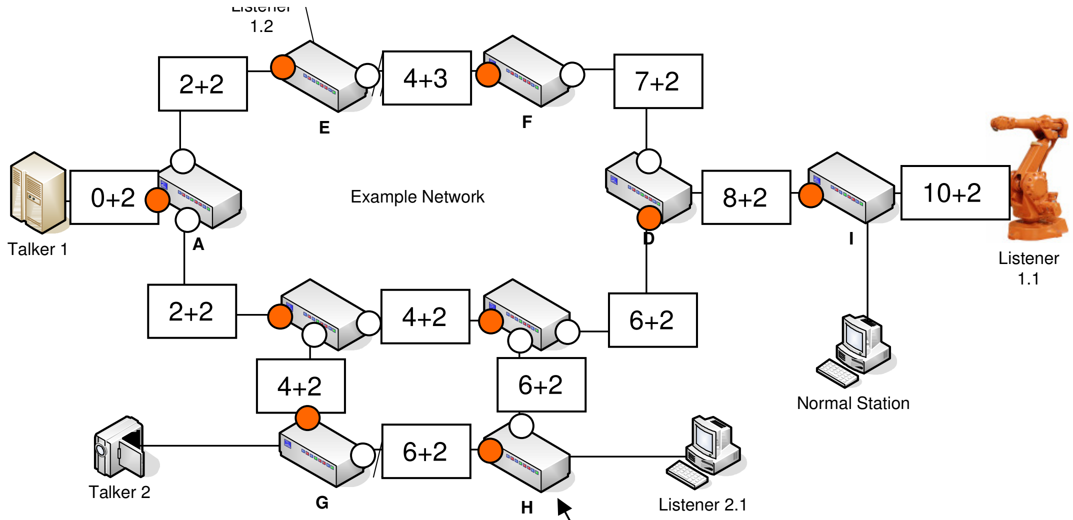
• Path selection could also be done on the basis of additional metrics, e.g. link „costs“

• In this case, this can augment/replace the previously discussed „arbitrary automatic“ and/or manual configuration methods

• Metrics could be parameters of special importance to specific application fields like e.g. network media

Philippe Klein‘s upcoming presentation on SRP multiple path selection will elaborate on that

simple, abstract example:

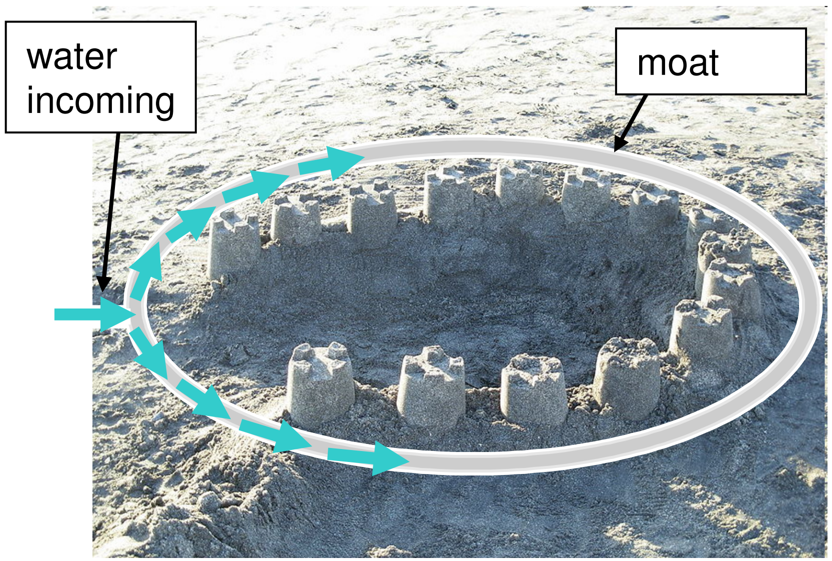


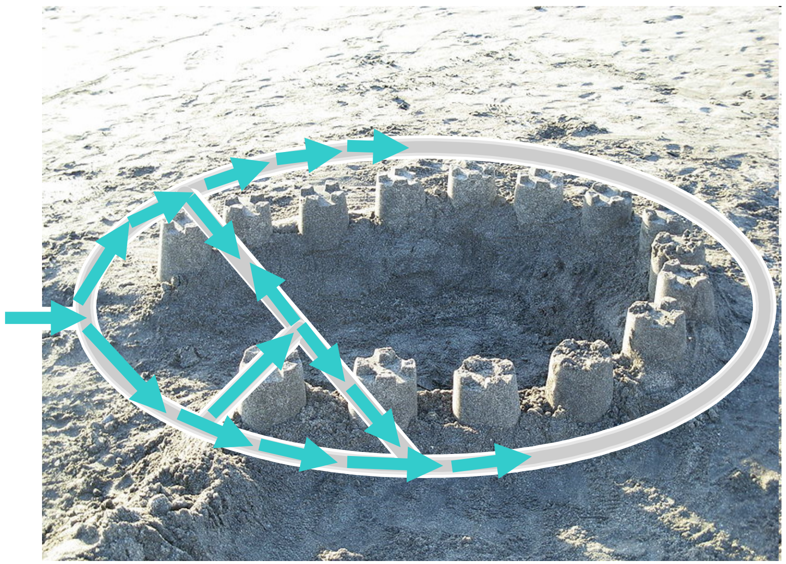
Tie-breaker needed…

The „moat model“

- The distribution of redundant streams thoughout the media-redundant network (registration and frame transmission) can be (figuratively speaking) regarded like water that enters a moat of a sand castle

- It is not entirely clear (if more complex moats for elaborate defensive purposes are designed), from where water will arrive at a certain point in the moat structure. But (given enough water is supplied), the whole moat will eventually be filled.



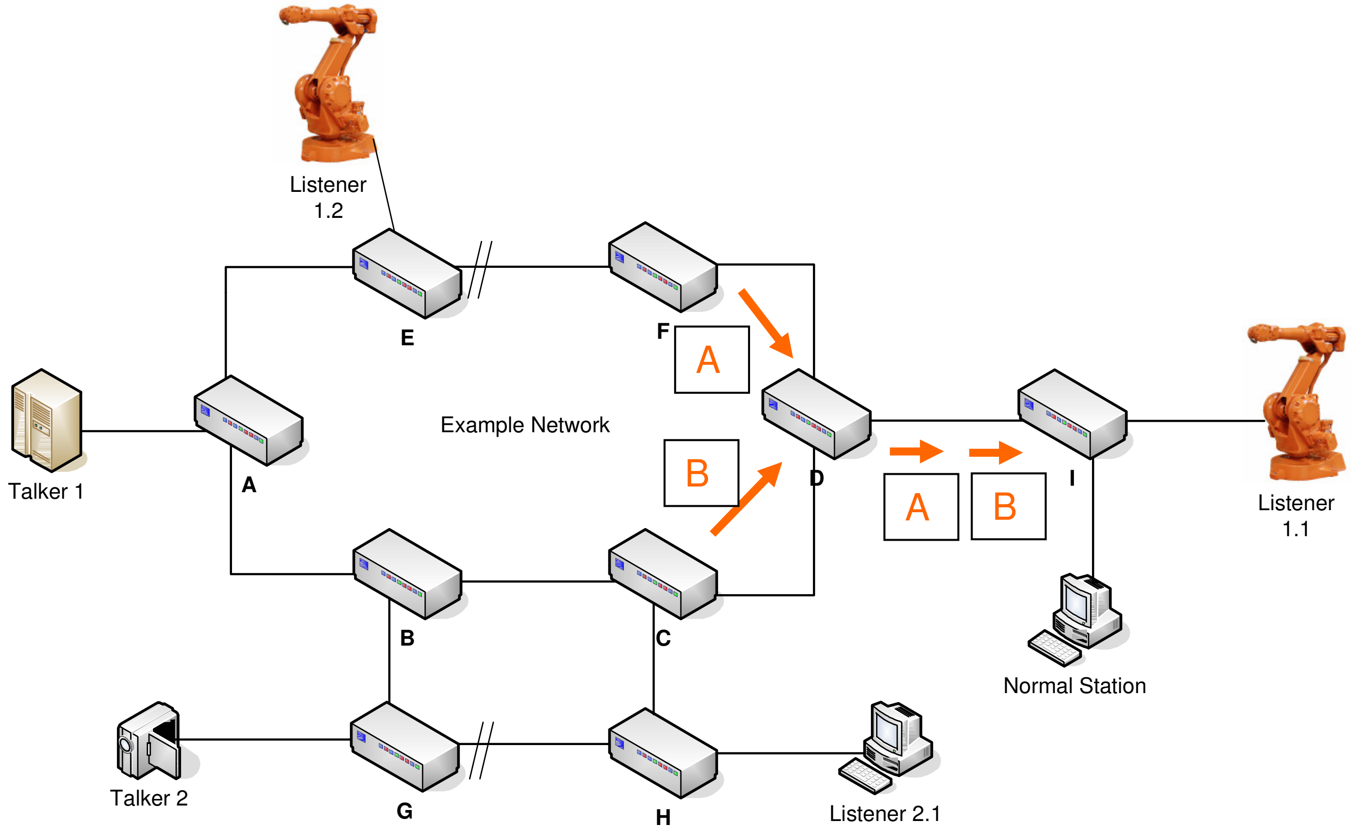


picture taken from wikipedia

What if we could model streams like the flows of water through a moat?

From fault-tolerance to load balancing?

Note: single points of failure included for explanatory purposes



- If a bridge forwards both frames, this method can also be used for load balancing purposes. (Makes more sense for a listener with two or more network interfaces)
- Bandwidth restrictions must be observed on shared links