```
package io.scalecube.cluster;

import io.scalecube.cluster.fdetector.FailureDetectorConfig;
import io.scalecube.cluster.gossip.GossipConfig;
import io.scalecube.cluster.membership.MembershipConfig;
import io.scalecube.transport.Address;
import io.scalecube.transport.TransportConfig;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Collections;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

/**
 * Cluster configuration encapsulate settings needed cluster to create and
successfully join.
 *
 * @see MembershipConfig
 * @see FailureDetectorConfig
 * @see GossipConfig
 * @see TransportConfig
 */
public final class ClusterConfig implements FailureDetectorConfig, GossipConfig,
MembershipConfig {

  // Default settings for LAN cluster
  public static final String DEFAULT_SYNC_GROUP = "default";
  public static final int DEFAULT_SYNC_INTERVAL = 30_000;
  public static final int DEFAULT_SYNC_TIMEOUT = 3_000;
  public static final int DEFAULT_SUSPICION_MULT = 5;
  public static final int DEFAULT_PING_INTERVAL = 1_000;
  public static final int DEFAULT_PING_TIMEOUT = 500;
  public static final int DEFAULT_PING_REQ_MEMBERS = 3;
  public static final long DEFAULT_GOSSIP_INTERVAL = 200;
  public static final int DEFAULT_GOSSIP_FANOUT = 3;
  public static final int DEFAULT_GOSSIP_REPEAT_MULT = 3;

  // Default settings for WAN cluster (overrides default/LAN settings)
  public static final int DEFAULT_WAN_SUSPICION_MULT = 6;
  public static final int DEFAULT_WAN_SYNC_INTERVAL = 60_000;
  public static final int DEFAULT_WAN_PING_TIMEOUT = 3_000;
  public static final int DEFAULT_WAN_PING_INTERVAL = 5_000;
  public static final int DEFAULT_WAN_GOSSIP_FANOUT = 4;
  public static final int DEFAULT_WAN_CONNECT_TIMEOUT = 10_000;

  // Default settings for local cluster working via loopback interface (overrides
default/LAN
  // settings)
  public static final int DEFAULT_LOCAL_SUSPICION_MULT = 3;
  public static final int DEFAULT_LOCAL_SYNC_INTERVAL = 15_000;
  public static final int DEFAULT_LOCAL_PING_TIMEOUT = 200;
  public static final int DEFAULT_LOCAL_PING_INTERVAL = 1_000;
  public static final int DEFAULT_LOCAL_GOSSIP_REPEAT_MULT = 2;
  public static final int DEFAULT_LOCAL_PING_REQ_MEMBERS = 1;
  public static final int DEFAULT_LOCAL_GOSSIP_INTERVAL = 100;
  public static final int DEFAULT_LOCAL_CONNECT_TIMEOUT = 1_000;

  public static final int DEFAULT_METADATA_TIMEOUT = 3_000;
  public static final String DEFAULT_MEMBER_HOST = null;
  public static final Integer DEFAULT_MEMBER_PORT = null;

  private final List<Address> seedMembers;
  private final Map<String, String> metadata;
  private final int syncInterval;
  private final int syncTimeout;
  private final int suspicionMult;
  private final String syncGroup;
  private final int metadataTimeout;

  private final int pingInterval;
  private final int pingTimeout;
  private final int pingReqMembers;

  private final long gossipInterval;
  private final int gossipFanout;
  private final int gossipRepeatMult;

  private final TransportConfig transportConfig;
  private final String memberHost;
  private final Integer memberPort;

  private ClusterConfig(Builder builder) {
    this.seedMembers = Collections.unmodifiableList(builder.seedMembers);
    this.metadata = Collections.unmodifiableMap(builder.metadata);
    this.syncInterval = builder.syncInterval;
    this.syncTimeout = builder.syncTimeout;
    this.syncGroup = builder.syncGroup;
    this.suspicionMult = builder.suspicionMult;
    this.metadataTimeout = builder.metadataTimeout;

    this.pingInterval = builder.pingInterval;
    this.pingTimeout = builder.pingTimeout;
    this.pingReqMembers = builder.pingReqMembers;

    this.gossipFanout = builder.gossipFanout;
    this.gossipInterval = builder.gossipInterval;
    this.gossipRepeatMult = builder.gossipRepeatMult;

    this.transportConfig = builder.transportConfigBuilder.build();
    this.memberHost = builder.memberHost;
    this.memberPort = builder.memberPort;
  }

  public static Builder builder() {
    return new Builder();
  }

  public static ClusterConfig defaultConfig() {
    return builder().build();
  }

  public static ClusterConfig defaultLanConfig() {
    return defaultConfig();
  }

  /** Creates cluster config with default settings for cluster on WAN network. */
```

```
  public static ClusterConfig defaultWanConfig() {
    return builder()
        .suspicionMult(DEFAULT_WAN_SUSPICION_MULT)
        .syncInterval(DEFAULT_WAN_SYNC_INTERVAL)
        .pingTimeout(DEFAULT_WAN_PING_TIMEOUT)
        .pingInterval(DEFAULT_WAN_PING_INTERVAL)
        .gossipFanout(DEFAULT_WAN_GOSSIP_FANOUT)
        .connectTimeout(DEFAULT_WAN_CONNECT_TIMEOUT)
        .build();
  }

  /** Creates cluster config with default settings for cluster on local loopback
interface. */
  public static ClusterConfig defaultLocalConfig() {
    return builder()
        .suspicionMult(DEFAULT_LOCAL_SUSPICION_MULT)
        .syncInterval(DEFAULT_LOCAL_SYNC_INTERVAL)
        .pingTimeout(DEFAULT_LOCAL_PING_TIMEOUT)
        .pingInterval(DEFAULT_LOCAL_PING_INTERVAL)
        .gossipRepeatMult(DEFAULT_LOCAL_GOSSIP_REPEAT_MULT)
        .pingReqMembers(DEFAULT_LOCAL_PING_REQ_MEMBERS)
        .gossipInterval(DEFAULT_LOCAL_GOSSIP_INTERVAL)
        .connectTimeout(DEFAULT_LOCAL_CONNECT_TIMEOUT)
        .build();
  }

  public List<Address> getSeedMembers() {
    return seedMembers;
  }

  public Map<String, String> getMetadata() {
    return metadata;
  }

  public int getSyncInterval() {
    return syncInterval;
  }

  public int getSyncTimeout() {
    return syncTimeout;
  }

  public int getSuspicionMult() {
    return suspicionMult;
  }

  public String getSyncGroup() {
    return syncGroup;
  }

  public int getMetadataTimeout() {
    return metadataTimeout;
  }

  public int getPingInterval() {
    return pingInterval;
  }

  public int getPingTimeout() {
    return pingTimeout;
  }

  public int getPingReqMembers() {
    return pingReqMembers;
  }

  public int getGossipFanout() {
    return gossipFanout;
  }

  public long getGossipInterval() {
    return gossipInterval;
  }

  public int getGossipRepeatMult() {
    return gossipRepeatMult;
  }

  public TransportConfig getTransportConfig() {
    return transportConfig;
  }

  public String getMemberHost() {
    return memberHost;
  }

  public Integer getMemberPort() {
    return memberPort;
  }

  @Override
  public String toString() {
    return "ClusterConfig{seedMembers="
        + seedMembers
        + ", metadata="
        + metadata
        + ", syncInterval="
        + syncInterval
        + ", syncTimeout="
        + syncTimeout
        + ", metadataTimeout="
        + metadataTimeout
        + ", suspicionMult="
        + suspicionMult
        + ", syncGroup='"
        + syncGroup
        + '\''
        + ", pingInterval="
        + pingInterval
        + ", pingTimeout="
        + pingTimeout
        + ", pingReqMembers="
        + pingReqMembers
        + ", gossipInterval="
        + gossipInterval
        + ", gossipFanout="
        + gossipFanout
        + ", gossipRepeatMult="
```

```
          + gossipRepeatMult
          + ", transportConfig="
          + transportConfig
          + ", memberHost="
          + memberHost
          + ", memberPort="
          + memberPort
          + '}';
  }

  public static final class Builder {

    private List<Address> seedMembers = Collections.emptyList();
    private Map<String, String> metadata = new HashMap<>();
    private int syncInterval = DEFAULT_SYNC_INTERVAL;
    private int syncTimeout = DEFAULT_SYNC_TIMEOUT;
    private String syncGroup = DEFAULT_SYNC_GROUP;
    private int suspicionMult = DEFAULT_SUSPICION_MULT;
    private int metadataTimeout = DEFAULT_METADATA_TIMEOUT;

    private int pingInterval = DEFAULT_PING_INTERVAL;
    private int pingTimeout = DEFAULT_PING_TIMEOUT;
    private int pingReqMembers = DEFAULT_PING_REQ_MEMBERS;

    private long gossipInterval = DEFAULT_GOSSIP_INTERVAL;
    private int gossipFanout = DEFAULT_GOSSIP_FANOUT;
    private int gossipRepeatMult = DEFAULT_GOSSIP_REPEAT_MULT;

    private TransportConfig.Builder transportConfigBuilder =
TransportConfig.builder();

    private String memberHost = DEFAULT_MEMBER_HOST;
    private Integer memberPort = DEFAULT_MEMBER_PORT;

    private Builder() {}

    public Builder metadata(Map<String, String> metadata) {
      this.metadata = new HashMap<>(metadata);
      return this;
    }

    public Builder addMetadata(String key, String value) {
      this.metadata.put(key, value);
      return this;
    }

    public Builder addMetadata(Map<String, String> metadata) {
      this.metadata.putAll(metadata);
      return this;
    }

    public Builder seedMembers(Address... seedMembers) {
      this.seedMembers = Arrays.asList(seedMembers);
      return this;
    }

    public Builder seedMembers(List<Address> seedMembers) {
      this.seedMembers = new ArrayList<>(seedMembers);
      return this;
    }

    public Builder syncInterval(int syncInterval) {
      this.syncInterval = syncInterval;
      return this;
    }

    public Builder syncTimeout(int syncTimeout) {
      this.syncTimeout = syncTimeout;
      return this;
    }

    public Builder suspicionMult(int suspicionMult) {
      this.suspicionMult = suspicionMult;
      return this;
    }

    public Builder syncGroup(String syncGroup) {
      this.syncGroup = syncGroup;
      return this;
    }

    public Builder metadataTimeout(int metadataTimeout) {
      this.metadataTimeout = metadataTimeout;
      return this;
    }

    public Builder pingInterval(int pingInterval) {
      this.pingInterval = pingInterval;
      return this;
    }

    public Builder pingTimeout(int pingTimeout) {
      this.pingTimeout = pingTimeout;
      return this;
    }

    public Builder pingReqMembers(int pingReqMembers) {
      this.pingReqMembers = pingReqMembers;
      return this;
    }

    public Builder gossipInterval(long gossipInterval) {
      this.gossipInterval = gossipInterval;
      return this;
    }

    public Builder gossipFanout(int gossipFanout) {
      this.gossipFanout = gossipFanout;
      return this;
    }

    public Builder gossipRepeatMult(int gossipRepeatMult) {
      this.gossipRepeatMult = gossipRepeatMult;
      return this;
    }

    /** Sets all transport config settings equal to provided transport config. */
    public Builder transportConfig(TransportConfig transportConfig) {
      this.transportConfigBuilder.fillFrom(transportConfig);
      return this;
    }

    public Builder port(int port) {
      this.transportConfigBuilder.port(port);
      return this;
    }

    public Builder connectTimeout(int connectTimeout) {
      this.transportConfigBuilder.connectTimeout(connectTimeout);
      return this;
    }

    public Builder useNetworkEmulator(boolean useNetworkEmulator) {
      this.transportConfigBuilder.useNetworkEmulator(useNetworkEmulator);
      return this;
    }

    /**
     * Override the member host in cases when the transport address is not the
address to be
     * broadcast.
     *
     * @param memberHost Member host to broadcast
     * @return this builder
     */
    public Builder memberHost(String memberHost) {
      this.memberHost = memberHost;
      return this;
    }

    /**
     * Override the member port in cases when the transport port is not the post to
be broadcast.
     *
     * @param memberPort Member port to broadcast
     * @return this builder
     */
    public Builder memberPort(Integer memberPort) {
      this.memberPort = memberPort;
      return this;
    }

    /**
     * Creates new clsuter config out of this builder.
     *
     * @return cluster config object
     */
    public ClusterConfig build() {
      if (pingTimeout >= pingInterval) {
        throw new IllegalStateException("Ping timeout can't be bigger than ping
interval");
      }
      return new ClusterConfig(this);
    }
  }
}
```